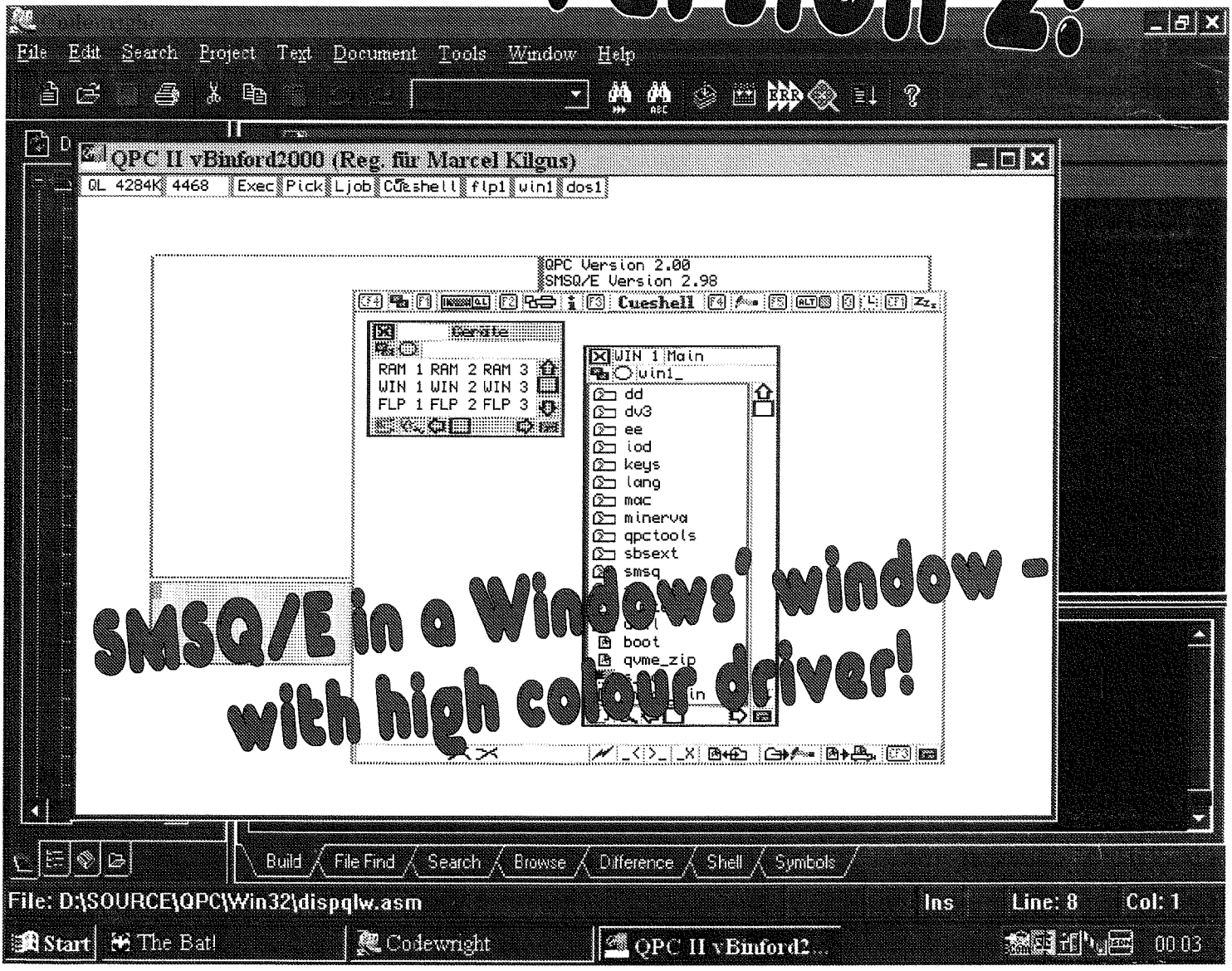
QL Today

Volume 5
Issue 2
July/August
2000

ISSN 1432-5454

The Magazine about QL, QDOS, Sinclair Computers, SMSQ...

QPC2 Version 2!



+ many Software Reviews
+ all series from last issue continue
+ Nasta starts explaining GoldFire

Contents

3 Editorial
4 News
5 ...and the winner is...
6 Updates & Corrections
7 Desperately seeking ...
John Rish
8 Email-List invitation
10 Gee Graphics! (on the QL?) Part 17
Herb Schaaf
13 Quanta/NESQLUG US QL Show
Al and Dorothy Boehm
17 The Wall - a game reviewed
Norman Dunbar
20 UNPICking LineDesign
Geoff Wicks
21 QPC2 ... the next Generation
Marcel Kilgus
23 You and Your Software - Just good Friends: Part 9 - Professional Blunders
Geoff Wicks
24 WXQT2 Review
Dilwyn Jones
27 Programming ProWesS in SBASIC - Part 2
Wolfgang Lenerz
34 BASIC Linker - a Review
Tim Swenson
38 Programming in Assembler - Part 8
Norman Dunbar
49 COSMOS - A Review
Brian Kemmett
51 Inside GoldFire - Part 1
"Nasta"
55 Byts of Wood
Roy Wood
57 Italian QL Show
BS QL Show Agenda

Advertisers

in alphabetical order

Jochen Merz Software 18, 19
Just Words! 29
QBOX USA 51
QBranch 8, 9
Quanta 37
RWAP Software 41
TF Services 45

QL Today

ISSN 1432-5454

German office & Publisher:

Jochen Merz Software
Im stillen Winkel 12
47169 Duisburg
Germany

Tel. +49 203 502011
Fax +49 203 502012
Box1 +49 203 502013
Box2 +49 203 502014
Mobile +49 1701 222277
email: JMerz@j-m-s.com
email: QLToday@j-m-s.com

English office:

Q Branch
20 Locks Hill
Portslade
BN41 2LB
United Kingdom

Tel. +44 1273 386030
Mobile +44 7836 745501
Fax +44 1273 381577
email: qbranch@qbranch.demon.co.uk
email: QLToday@j-m-s.com

Editor:

Dilwyn Jones
41 Bro Emrys
Tal-Y-Bont, Bangor
Gwynedd
United Kingdom LL57 3YT

Tel. +44 1248 354023
email: dilwyn.jones@dj.softnet.co.uk
email: QLToday@J-M-S.com

Co-Editor:

Bruce Nicholls
57 Shaftesbury Rd
Romford
Essex RM1 2QJ
United Kingdom

Tel +44 1708 755759
Fax +44 870 0568755
email: qltoday@q-v-d.demon.co.uk
email: QLToday@j-m-s.com

QL *Today* is published bi-monthly, our volume begins on beginning of June. Subscriptions begin with the current issue at the time of sign up. Please contact the German or English office for current subscription rates.

We welcome your comments, suggestions and articles. YOU make **QL *Today*** possible. We are constantly changing and adjusting to meet your needs and requirements. Articles for publication should be on a 3.5" disk (DD or HD) or sent via Email or into one of the JMS-BBS's. We prefer ASCII, Quill or text87 format. Pictures may be in _SCR format, we can also handle GIF or TIF or JPG. To enhance your article you may wish to include Saved Screen dumps. PLEASE send a hardcopy of all screens to be included. Don't forget to specify where in the text you would like the screen placed.

Article and Advertising deadlines are as follows:

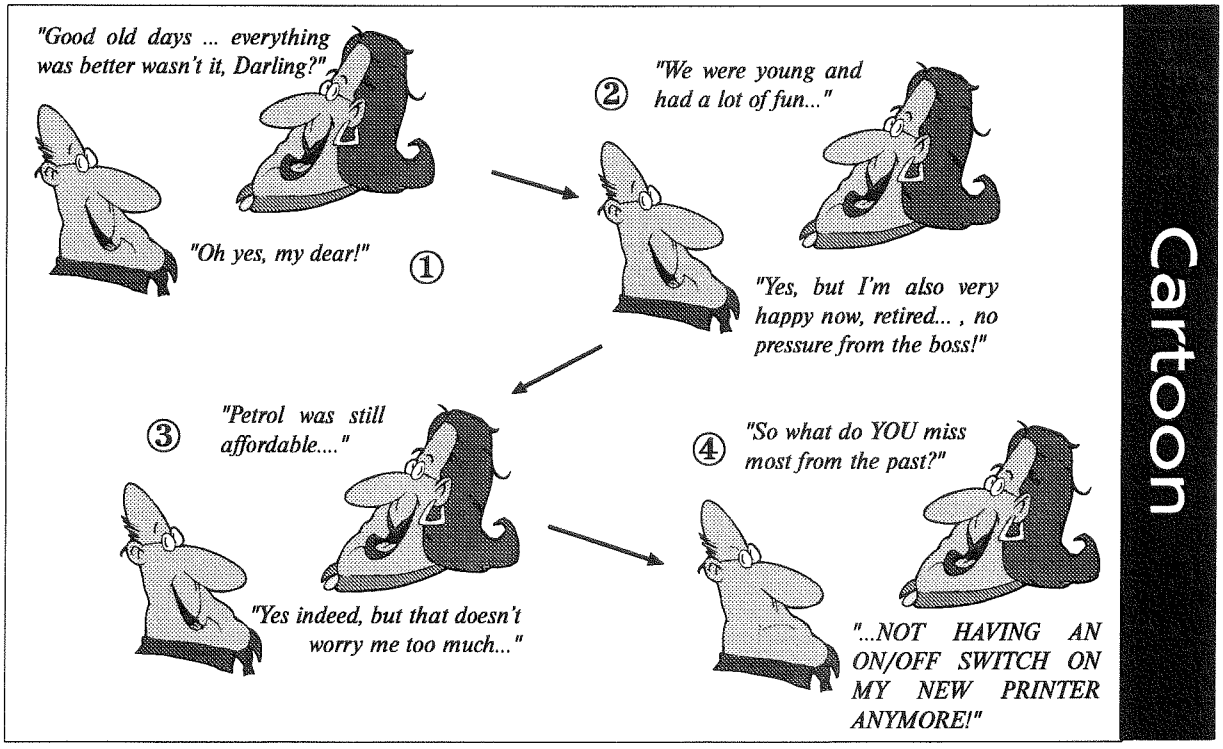
Issue 1: 30 April	Issue 2: 30 June
Issue 3: 30 August	Issue 4: 30 October
Issue 5: 30 December	Issue 6: 28 February

QL *Today* reserves the right to publish or not publish any material submitted. Under no circumstances will **QL *Today*** be held liable for any direct, indirect or consequential damage or loss arising out of the use and/or inability to use any of the material published in **QL *Today***. The opinions expressed herein are those of the authors and are not necessarily those of the publisher.

Editorial

Dilwyn Jones

Once again my thanks are due to our regular band of contributors who have provided a varied and interesting selection of articles for this issue - without their help, this magazine would not be possible. And of course, thanks are due to the editorial team of Jochen, Roy and Bruce, without whose help this issue would not have been possible at all because of my recent overwhelming job pressures and time taken to prepare for a big event in my life in September (no, I don't mean I'm buying a Q40). Speaking of big events, regular readers will know I am an avid user of the QL emulator QPC. So far, it has been one of the few SMSQ/E platforms not to receive the "colour drivers" treatment. Well, Marcel Kilgus (author of QPC) has been hard at work on QPC2, and we managed to drag him away from it long enough to write an article about it. What is happening to QPC2? Read his article to see. Jon Dent is also beavering away on the QDOS TCP/IP project and there is a further progress report elesewhere in this issue.
Norman Dunbar was asked to reveiew a copy of The Wall, a new QL game ›from Wolfgang Lenerz. This nearly proved to be a mistake, as he got so addicted to it he barely stopped playing it long enough to write for this issue - see his review of this rather unique game! Wolfgang Lenerz in turn writes about Prowess in this issue, while Nasta brings us some information about his Goldfire hardware project.
Will you be looking forward to Quanta's QL2000 meeting this autumn? If not, you'll be in a minority, for everyone seems to be talking about this biggest of QL events at the moment, with interest in attending shown from QLers worldwide! Although after my big event in September, it'll be perhaps the second biggest event of the year for me...



NEWS

RWAP Software / Rich Mellor

QL Cosmos is now version 2.02 and handles large screens much better, using as much of the screen as possible. Updates from the original cost £5.00

Q-Help is now version 1.05 which overcomes the problems with case sensitivity mentioned in the review. It also includes the Pointer Environment files (it always did!) and enables the user to search for help on a given keyword from within the program without searching through the list for it. Updates to the latest version cost £1 and an SAE.

Due to popular demand we have now released Q-Index v1.04 as a companion program to Q-Help for £5. This is the index program supplied with the SBASIC/SuperBASIC Reference Manual and allows you to enter a topic (such as Battery Backed Clocks) and receive a list of all relevant keywords. You can then select a keyword to launch Q-Help for information.

QL Cash Trader (which was sold as Trading Accounts in the past) is now v3.7. Upgrades from the original cost £5, upgrades from earlier RWAP Software versions cost £1 and an SAE

QL Genealogist is now v3.25 - this is well worth the update, with the ability to use FileInfo II to launch other programs (such as Quill) to read/amend documents linked with an individual. It will also use QMenu if this is available to find filenames, as well as passing the name of the tree to be loaded when the program first starts up (useful for FileInfo II). Other improvements include the ability to alter more parameters (for the printer and for Age difference tolerances) and being able to generate a list of persons either alphabetically or by Network ID. There are also a few minor bug fixes. Upgrades from earlier RWAP Software versions cost £1 plus SAE.

I also have a range of second hand items for sale - please send an SAE for latest list.

Paragraph News / Francois Lanciault

This is to notify the community that the free version of Paragraph has been updated to version 1.07. Many bugs are now things of the past since v1.04 that was available on the internet. The archive should be available soon on Thierry site and the PROGS site.

Also version 2.03 of the program has been sent to all registered users with email. If you are a registered users and did not received the update by email please let me know.

Version 2.03 now has HTML export facility. The program can generate complex HTML documents with tables, multi-column text, images, various font and fontsize etc.

Dilwyns Website

I have now updated my website to include the latest version of Dave Westbury's Photon JPEG viewer program, which now recognises GD2 mode 32 (QXL and forthcoming QPC 64k colours mode).

Recent additions to the same site include a range of programs from Mark Knight (K-Base, Molecular Graphics, Heartbeat, Popcalc, Q-Page etc) and from Norman Dunbar (former DJC programs including Winback hard disk backup, The Gopher file search utility, DJToolkit of BASIC extensions and a few other freeware programs). Norman has now made these ex-DJC programs into freeware and I am very grateful to him for that.

These programs may be found on the Other Software page - from my homepage select the 'QL Software' page then click on the link 'click here to visit the Other Software page'. My homepage is at

http://www.soft.net.uk/dj/index.html

To go direct to the Other Software page go to **http://www.soft.net.uk/dj/software/other/other.html**

Beginners Club, Italy - New Email Address

On 15th July 2000 the old e-mail address: **beginners@geocities.com** will expire.

Since today you can use the new address: **beginners.geo@yahoo.com**

Doc2html / Dilwyn Jones

An early Beta test release of my DOC2HTML program, allowing simple HTML pages to be created in Quill (including facilities for images and links inclusion, is now available from my Web site. Create simple Web pages in Quill, save them as DOC file then use this program to convert them to HTML.

Do bear in mind that this is a Beta test version, so it may not work on all platforms or may have bugs lurking in there!

Try pointing your browser at

http://www.soft.net.uk/dj/software/freeware/freeware.html

and scroll down to the DOC2HTML section to download the program.
Any feedback on this program would be appreciated. While you are not going to be creating the flashiest web site ever using this program, it does afford a means for those unfamiliar with HTML to design a first basic web site. The program does not do any FTP to upload the pages of course, you'll have to rely on whatever platform you currently use to access the net for that until full QL internet access and ftp comes along.

G. Plavecs Website has moved

This may be of interest for your bookmarks:
My web site has been moved to
http://kuel.ctw.cc/
(previously www.altern.org/kuelat)
The good news is that I have now more room for ql things. The bad news is that I have to update some of the pages.
You can also find a QXL.WIN format description
http://kuel.ctw.cc/qxlwin.html
Gerhard maintains an Austrian QL website, so you should be able to find more information about the Austrian QL show here too!

QL Users Email Database

Robin Barker has re-programmed the QL Users Email Database. This is a list of all QL Users who are online complete with their email addresses. If you visit the Quanta site
www.quanta.uni.cc
and find that you are not mentioned (and would like to be) there are instructions on how to be put onto it.
In the course of this he had to use CGI programming and, in his communication to this magazine, he went into a short rant at this point decrying the need to have to keep learning more computer languages.
The Quanta site now contains a large database with further information on the users volunteered by the users themselves. The website does have the ability to 'bulk email' groups of addresses via a built in search engine. Those of you who have Internet capability are invited to visit it.

Jochen Merz Software

For major news, read the QPC2 article from Marcel. I hope to have version 8 of the Menu Extension ready soon. No details can be given here, no space. As there seem to be confusion about the naming (people refer to it as "QMenu"), I will rename it to "QMenu" soon and "QMenu" will be called "Menu Extension Programmers manual or something similar.

...and the winner is...

We have received many emails after we set the deadline in the previous issue, but fortunately no postcards arrived after the deadline.

So, here is the "official" result:

Dietrich: Digital Clock	12%
Duncan: Q40 and Aurora...	15%
Marcel: Star3D	23%
Per: Blocks	15%
Stephen: Kaleidoscope ...	35%

Congratulations, Stephen, the prize will be on its way to you!

Other interesting replies: 86% of the readers which replied by postcard are able to handle HD disks. This is good to know, but we will stick to DD disks in the future to satisfy all readers - and make only exceptions where HD disks are essential to run the disks contents (i.e. it would be of no use on DD anyway, like ProWesS was).

Most readers would like to see cover disks with demo versions of commerical software, which is something we always wanted to do. Now, that we have confirmation that this is something also demanded by our readers, expect one of our next cover disks to contain demo versions.
Call to commercial software authors: please inform QL Today about the availability of demo versions (preferably via email, so that we can easily contact you when we need to prepare the cover disk).
There seems to be a demand for PD utilities in general - which could be found on many of our previous cover disks. Pretty good to see that these are preferred items on our cover disks - we will carry on providing you with the most useful tools.

We would like to say "Thank you" to all participating authors for their screen savers, and also to all readers who returned the postcards for voting and informing us what you would like to see in the future, and for ensuring us we are doing something you like to read/test.

UPDATES & CORRECTIONS

OOPS...
In the last issue, we published a news item about Daniel Baum's Website. Unfortunately, we managed to print the old address. The correct address for his website is:
www.qldesign.com

OOPS (2)...
In the editorial column of the last issue, I referred to a new version of the "colour drivers" for the QL. In fact, what I meant to say was that it was for the QXL. Apologies for the error.
Dilwyn Jones

Wolfgang Uhlig wrote:
Hi Geoff,
I read your article in the latest QL Today with interest. There is one point where you are not well informed I suppose. To get a list of all items of an EASYPTR menu use the command MLIST in the following way:

```
OPEN #3,con
OPEN_over #4,ram1_mymenu_ref
MDRAW #3,mymenu
MLIST #4,#3
CLOSE #4,#3
```

Then you will have a wonderful list of all the items with size, place, contents etc. in the file "ram1_mymenu_ref".

Graham Wall wrote:
I've been trying to download Dave's photon_zip program [from Dilwyn's website] Are you sure the address you have given in QL To-Day is correct. Every combination of soft.net; softnet; dj first or after produce a ziltch on the internet connection I have here in Spain.
I am not sure if Graham is on this mailing list, however it will serve to point out an error in the latest QL Today. A directory name has been omitted in the website address published on page 5 of Vol 5 Issue 1 of QL Today, which has left a number of people wondering how to get hold of the Photon JPG handler program by Dave Westbury.
The correct URL for the page concerned is:
www.soft.net.uk/dj/software/other/other.html
Apologies to Dave Westbury (the author of Photon) for this error.

John E. Juergens sent us a useful tip:
Recently, I was browsing through the MicroSoft site and noticed that DirectX 7A was available, so, I downloaded and executed it.
I may be wrong but I have the distinct impression that it noticeably increased the speed at which my QPC2 operated.
Just thought I'd mention it in case you hadn't noticed the upgrade.
Thanks - no, I haven't noticed - but I'm not surfing to the MicroSoft website. But now I'll try it myself.

Jon Dent keeps us up-to-date:
Hi Folks this is an updated progress report for my TCP/IP project.

Working:
- System ... extended for TCP
- SOQL (socket library) ... updated with TCP functions
- UDP
- IP
- SLIP
- TCP ... at last!

Work in progress:
- TCP ... Undergoing thorough tests
- POP3 ... Really just to test TCP but can be used to collect emails from a POP3 server.

There are probably lots of (small) details which need to be added to achieve full RFC compliance. The stack is working on my SMSQ/E SGC sHermes system but I got some funnies when I tried it on Minerva which I haven't had time to investigate.

Built but not yet integrated:
PPP ... I'd like to have the rest of the stack running stably and beta tested using SLIP before integrating PPP.

Planned:
- Documentation!!!
- SMTP (application email)
- Trap #2/#3 support. (system)
- DNS (application/stack)

Thinking about:
- TELNET
- FTP
- WWW

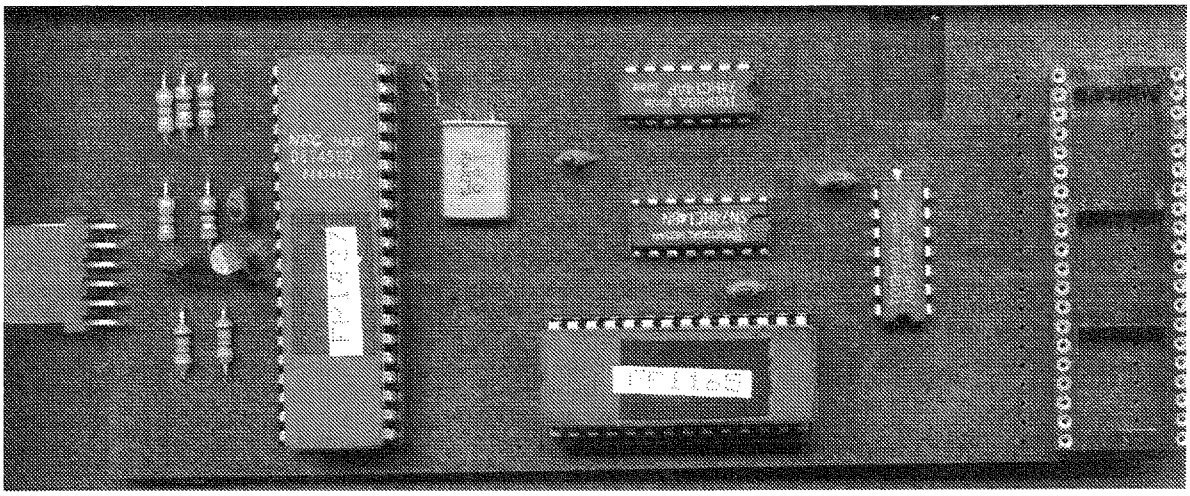
...And now a question to everyone using a dial-in ISP: Does your ISP support SLIP? Sometimes they do but don't advertise it. You just have to enter SLIP instead of PPP when logging in. Others don't give you the chance to enter anything they just switch to PPP.(Maybe you could check your login script?)

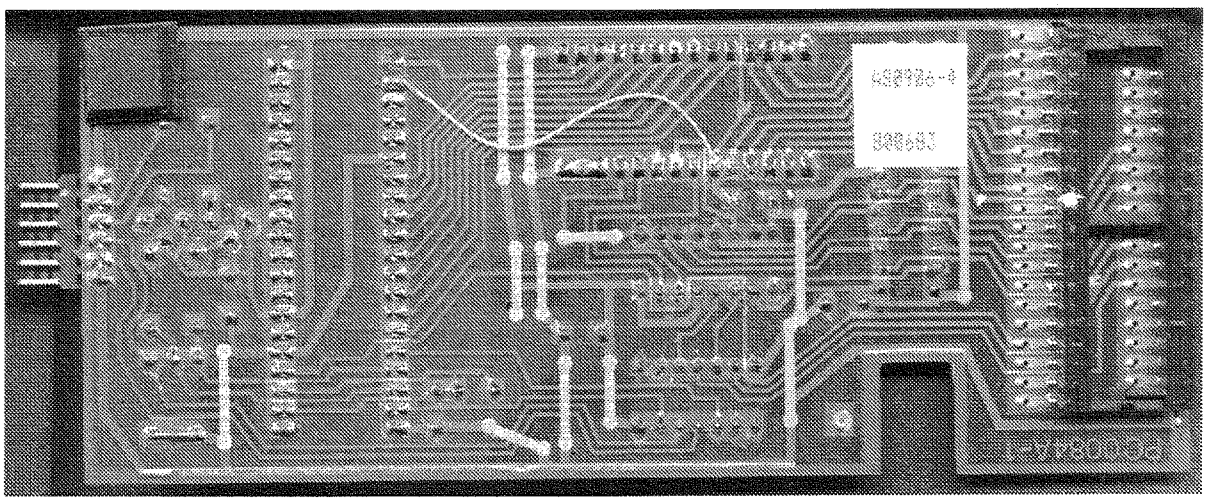
Desperately seeking ...

John Rish

I have a keyboard interface (I think) which is not a Di-Ren, and looks home made, and also very professional. I am trying to find out what it is, can anyone help with the description below?
It has two EPROMs with labels on them, (typed on a dot matrix printer) FP1427, and FP1165.
Three 16 pin IC chips, two each 74HC14ap, and one SN74hC148N. A place to put the 8049, a crystal, a couple of transistors, several resisters/-capacitors, on one end it has a connector that has six pins that connect to a six pin din plug. The circuit board is 6 inches by 2 1/2 inches. Does this sound like anything you have seen before.

Tony Firshman does not recognise it, QBranch has not seen it either, nor has Jochen Merz.



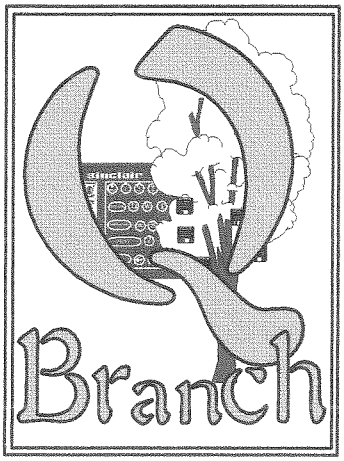


Email-List invitation

If you returned the postcard which we added to Issue 6 of Volume 4 and you filled in your email address and ticked the field "please add me to the QL News Email list", then you should be on the list by now, provided, your email address was correct.
You should have received an invitation via email which you hopefully returned, otherwise you are not on the list. This security feature makes sure that you do not get into lists you do not want to be on, because somebody else put you on the list without your knowledge.

If you are not on the list, you can get yourself easily into it: go to
www.j-m-s.com/smsq/index.htm
scroll down a bit and you will find a link to the form where you can enter your email address. Same procedure: you will receive an email which you have to return.
Do not worry: you will not receive many mails (e.g. feel "spammed"). We only send mails in case of real news, or to announce QL shows etc. This can happen twice or three times a month, and sometimes a month is completely quiet.
And, of course, you can always unsubscribe yourself from the list. So why not join now?

Q Branch



PROGRAMMING

QD 98	£ 53.00
QD + QBasic	£ 69.00
QD + Qliberator + QBasic	£ 110.00
Qliberator	£ 50.00
Master Spy v 3.3	£ 30.00
QPTR	£ 37.00
Easyptr pt 1 & 2 (together)	£ 33.00
Easyptr pt 3 (C library)	£ 16.00
QMake	£ 18.00
QMon / JMon	£ 22.00
Basic Linker	£ 22.00
DISA 3	£ 37.00
QMenu	£ 16.00

Text 87

£ 79.00

Typset 94 £ 29.00
Fountext 94 £ 39.00
2488 drivers £ 29.00
Epson ESC/P2 drivers £ 26.00

Text 87 is the only QDOS / SMSQ wordprocessor capable of handliing the full screen on the Aurora / QXL / QPC systems. New drivers are currently being written.

Following the closure of the shop in Worthing Q Branch have an amount of stuff available for sale. We have powered speakers from £5.00 each, many power and loop through cables (£2.00), printer cables (£2.70), mouse mats (75pence), wrist rests (£1.50), mice from £3.00, 486 processors from £5.00 and a host of other stuff. There is a superb full tower case with powers supply and enough space to install an Aurora and a Q40 for only £ 25.00. I would really like to clear all of this from my house so, if you need anything at all please call us.

On a more QL oriented note we also have some second user QL stuff including QXLs, Super Gold Cards Auroras, Qubides, ROMdisqs and superHermes. The colour drivers for QPC 2 should be available soon. Call us for details . I hope to see many of you at the LOndon Show in September.

'Just Words' by Geoff Wicks

THESAURUS, STYLE CHECK, SOLVITPLUS 3

£ 10.00 ea / ANY 2 PROGRAMS £ 18.00 / ALL 3 PROGRAMS £ 25.00
(Includes Pointer and non-pointer driven versions)
(P.E. versions need Hot_rext, WMAN and PTR_GEN or SMSQ/E to run)
Upgrades from previous versions £ 2.50 + S.A.E. New Manuals £ 1.50

QL2PC - New program !!

Convert text files from QL to PC formats and much more !

Only £ 10.00 *Now with HTML support !*

Spelling Crib : PD program £ 1.50 +SAE
or Free if you buy all three programs

UTILITIES

FiFi 2	£ 22.00
QSup	£ 32.00
QSpread v2.04	£ 66.00
Cueshell 2	£ 30.00
Qload / Qref	£ 15.00
Disk Mate 5	£ 16.50
QPAC 1	£ 20.00
QPAC 2	£ 40.00
QTYP 2	£ 30.00
QLQ	£ 32.00

We are currently out of stock of the SuperBasic Reference Manual Place your order now to get one as soon as it is reprinted

The SBASIC / SuperBASIC Reference Manual

The complete definitive guide to BASIC programming in QDOS / SMSQ including three disks of PD toolkits, example procedures and an electronic index.
compiled by Rich Mellor, Franz Hermann and Peter Jaeger

Over 500 pages !
£ 40.00
+ postage

Tel +44 (0) 1273-386030 fax +44 (0) 1273-381577
Mobile +44 (0) 7836-745501
email qbranch@qbranch.demon.co.uk web : http://www.qbranch.demon.co.uk

Q Branch

Feeling out on a limb ?
Reach out for Q Branch.
Suppliers of Quality QDOS/SMSQ products
Hardware and Software.

20 LOCKS HILL, PORTSLADE,
E. SUSSEX. BN41 2LB. UK.

Please Note our new address and phone numbers

ProWesS

ProWesS (now free !)	£ 1.60
DATAdesign	£ 24.00
Fontutils	£ 30.00
File Search	£ 12.00
PFlist	£ 12.00
Dilwyn's Fontpack	£ Call
LINEdesign v 2.16	£ 24.00
PWfile	£ 18.00

Paragraph

The ProWesS word processor

¶ Demo version £ 1.50 + postage
Full Registered version £ 18.00

Version 2.03 available now !

Hardware

We have a small stock of second user items. Auroras / Qubides / Gold Cards / Qplanes / superHermes etc. call us to get details of the items available. These are going fast so call soon.

Super Gold Cards	£ 125.00 Few only
QXL	£ Call
Recycled superHermes	£ 80.00 *
Recycled Gold Card	£ 60.00 *
Recycled Aurora	£ 75.00 *
Aurora	£ 90.00
Qubide	£ 55.00
Qplane	£ 25.00
Aurora cables	£ 3.00
Aurora rom adaptor	£ 3.00
'Arfa Braquet'	£ 8.00
'Son of Braquet'	£ 18.00
The 'Braquet'	£ 16.00
MC plate	£ 6.50

* when available.
14" and 15" monitors for the Aurora - Call.

SMSQ/E

Gold Card / Atari / QXL Version

£ 76.00

Various Atari versions : call for details

The Colour Drivers are running on the Q Branch Q 40 !
Coming to a computer near you in 2000!

QPC 2 is here !

£ 90.00 (£ 70.00 SMSQ/E Owners)

Upgrades from QPC 1 £ 30.00
(return master disk)

Special offer !
Get Cueshell for only £15.00 with any copy of SMSQ/E

Q Branch Programs

The Knight Safe 3 *NEW VERSION!*	£ 35.00
upgrades from previous versions	£ 5.00
Q - Route v1.08C Route finding programme	£ 25.00
Q - Count *NEW VERSION!* Pointer driven home accounting	£ 25.00

The Fractal Collection !

This is a brand new program which will produce stunning animated fractal patterns. It will run on anything from a Gold Card to the Q40 and will be capable of using the power of the colour drivers when they are released. Complete with many example files and routines to design your own screens.

Only £ 35.00

There are so many extras available for the Q 40 that it is hard to list them here. If you are interested in one of these boards than contact us and we will supply the details. the basic prices are shown here.

Q 40 board with 16Mb RAM & I/O card	£ 330.00
+ SMSQ/E	£ 30.00
Extra 16 Mb RAM	£ 30.00
Tower Case	£ 45.00

All prices do not include shipping.

We can accept payment by VISA, Mastercard and Switch. You can also pay by Eurocheques made out in Sterling or a Sterling cheque drawn on a UK Bank. Prices include Post and Packing in Europe.

Gee Graphics! (on the QL?) Part 17

Herb Schaaf

3 Views added - Front, Top, and Side

In GG#16 we presented one way of finding the solution for the distance and direction from a Point to a Line in 3-space. To "see" the results graphically you can merge the listing "3Views_added" to the previous listing "Point_to_Line_3D_bas". Use quotation marks around the filename when you merge since it starts with a numeric instead of a alphabetic character. An error-trapping REPeat check_angle has been included.

After the usual entering of data with numerical solution as before, you will then have an added graphical display in three orthographic views. A Front view in the lower left window, a Top view in the upper left window, and a right Side view in the lower right window. The upper right window will identify each of the points, lines, etc. as used in the algorithm. To "see" the steps of the algorithm in sequence just touch the spacebar each time the program reaches PAUSE.

Next time? Perhaps we can have a look at matrices. They seem to be quite useful in computer graphics.

Listing "3Views_added"

```
100 REMark merge this 3Views_added to Point_to_Line_3D_bas
110 REMark HL Schaaf June 13, 2000
120 REMark to go with GG #17 and GG #16
210 views_in_3D
1885 REPeat check_angle
1915 IF ((dirang(axis) >= min_ang) AND (dirang(axis) <= (180-min_ang))) THEN
1916   EXIT check_angle
1917 END IF
1918 END REPeat check_angle
2860 PRINT \\\\,,"Touch any key for 3 views"
2870 REMark PRINT \\\\,,"Touch any key to exit"
2930 :
2940 DEFine PROCedure views_in_3D
2950 REMark what limits for values ?
2960 max = 0 : min = 1E6
2970 DIM min_max(3,2)
2980 FOR i = 1 TO 3
2990   IF line3d(1,i) < min_max(i,1) : min_max(i,1) = line3d(1,i)
3000   IF line3d(1,i) > min_max(i,2) : min_max(i,2) = line3d(1,i)
3010   IF line3d(2,i) < min_max(i,1) : min_max(i,1) = line3d(2,i)
3020   IF line3d(2,i) > min_max(i,2) : min_max(i,2) = line3d(2,i)
3030   IF Pt_in_3D(i) < min_max(i,1) : min_max(i,1) = Pt_in_3D(i)
3040   IF Pt_in_3D(i) > min_max(i,2) : min_max(i,2) = Pt_in_3D(i)
3050   IF Ft_of_Pt2Line(i) < min_max(i,1) : min_max(i,1) = Ft_of_Pt2Line(i)
3060   IF Ft_of_Pt2Line(i) > min_max(i,2) : min_max(i,2) = Ft_of_Pt2Line(i)
3070   min_max(i,0)= min_max(i,2) - min_max(i,1)
3080   IF min_max(i,2)> max : max = min_max(i,2)
3090   IF min_max(i,1)< min : min = min_max(i,1)
3100   min_max(0,0) = max - min
3110 END FOR i
3120 REMark greatest range in any case is min_max(0,0)
3130 REMark set same scale for all views, but may have differing origins
3140 :
3150 MODE 4
3160 WINDOW #0,256,128,256,0  : REMark for prompts, etc.
3170 WINDOW#1,256,128,256,128 : REMark Right side view
3180 WINDOW#2,256,128,0,0     : REMark Top view
```

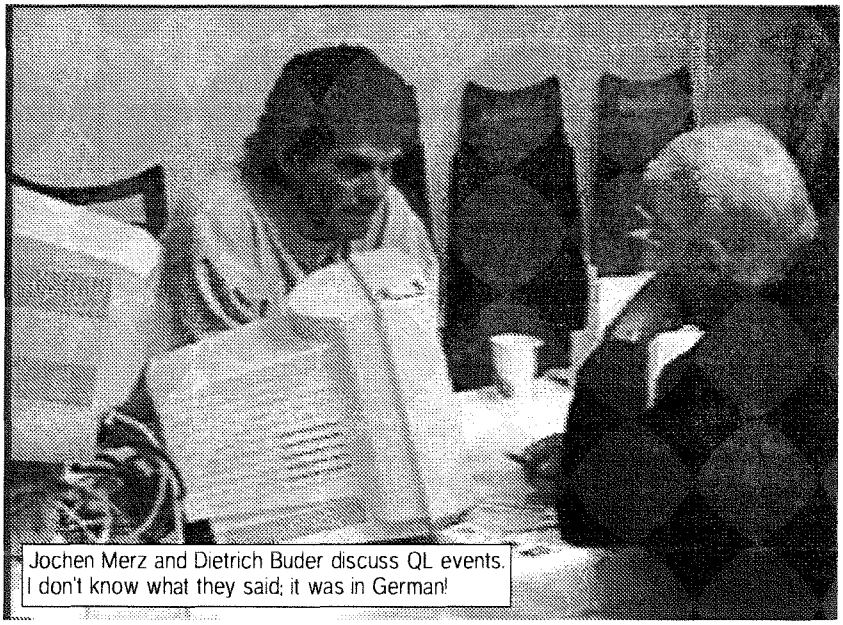
```
3190 OPEN #3,con_256x128a0x128_128 : REMark frontal view
3200 DIM h(3) : DIM v(3) : REMark horizontal and vertical axes
3210 h(1) = 3 : h(2) = 1 : h(3) = 1
3220 v(1) = 2 : v(2) = 3 : v(3) = 2
3230 FOR i = 0 TO 3
3240   PAPER #i,0
3250   INK#i, 7
3260   BORDER#i,1,i*2
3270   CSIZE #i,0,0
3280   CLS#i
3290 END FOR i
3300 INK#0,4
3310 BORDER #0,1,2
3320 range = min_max(0,0)*1.1 : REMark add 10%
3330 margin = min_max(0,0)*5E-2 : REMark 5% at edges
3340 FOR i = 1 TO 3
3350   SCALE#i, range, (min_max(h(i),1) - margin), (min_max(v(i),1) - margin)
3360   CLS#i
3370 END FOR i
3380 DIM locat$(3,5)
3390 locat$(1)=" Side"
3400 locat$(2)="  Top"
3410 locat$(3)="Front"
3420 DIM ax$(3,2) :REMark h then v
3430 ax$(1) = "ZY"
3440 ax$(2) = "XZ"
3450 ax$(3) = "XY"
3460 FOR i = 1 TO 3
3470   LINE #i, 0,range - (2 * margin) TO 0,0 TO range,0
3480   AT #i, 7,35 : PRINT #i;locat$(i)
3490   AT #i, 8,35 : PRINT#i;" View"
3500   AT #i, 11,37:PRINT #i;ax$(i,1);" ";CHR$(189)
3510   AT #i, 0,2 : PRINT #i;ax$(i,2)
3520   AT #i, 1,2 : PRINT#i;CHR$(190)
3530 END FOR i
3540 :
3550 REMark show point 1
3560 CLS #0
3570 FOR p = 1 TO 2
3580   PRINT #0\"Point ";p;" of Line ";
3590   INK#0, 2*p
3600   PRINT #0,"o"
3610   INK#0,4
3620   FOR i = 1 TO 3
3630     INK #i, 2*p
3640     POINT#i, line3d(p,h(i)),line3d(p,v(i))
3650     CIRCLE#i, line3d(p,h(i)),line3d(p,v(i)), range/100
3660   END FOR i
3670   PAUSE
3680 END FOR p
3690 :
3700 REMark draw line between points in red after
3710 REMark producing line in white
3720 REMark solve end points of line from min to max ?
3730 PRINT #0\"Line in space ";
3740 INK #0,238
3750 PRINT #0,"_______"
3760 INK #0,4
```

```
3770 FOR i = 1 TO 3
3780   INK # i, 6
3790   POINT #i, line3d(1,h(i)),line3d(1,v(i))
3800   LINE TO#i, line3d(2,h(i)),line3d(2,v(i))
3810   LINE TO #i, Ft_of_Pt2Line(h(i)),Ft_of_Pt2Line(v(i))
3820   INK#i,2
3830   POINT #i, line3d(1,h(i)),line3d(1,v(i))
3840   LINE TO#i, line3d(2,h(i)),line3d(2,v(i))
3850 END FOR i
3860 PAUSE
3870 :
3880 REMark show point 3 in space
3890 :
3900 PRINT #0\"Point 3 in space ";
3910 INK #0,6
3920 PRINT #0,"o"
3930 INK#0,4
3940 FOR i = 1 TO 3
3950   INK #i, 6
3960   POINT#i, Pt_in_3D(h(i)),Pt_in_3D(v(i))
3970   CIRCLE#i, Pt_in_3D(h(i)),Pt_in_3D(v(i)),range/100
3980 END FOR i
3990 PAUSE
4000 :
4010 PRINT #0\"Point 1 to Point 3 ";
4020 INK#0,242
4030 PRINT #0,"-------"
4040 INK#0,4
4050 FOR i = 1 TO 3
4060   INK #i , 242
4070   POINT#i, line3d(1,h(i)),line3d(1,v(i))
4080   LINE TO #i, Pt_in_3D(h(i)),Pt_in_3D(v(i))
4090 END FOR i
4100 PAUSE
4110 :
4120 PRINT #0\"Offset along Line to foot ";
4130 PRINT#0,"--------"
4140 FOR i = 1 TO 3
4150   INK #i, 4
4160   POINT #i, line3d(1,h(i)),line3d(1,v(i))
4170   LINE TO #i, Ft_of_Pt2Line(h(i)),Ft_of_Pt2Line(v(i))
4180 END FOR i
4190 PAUSE
4200 :
4210 PRINT#0;"Shortest distance from Point to Line ";
4220 INK#0,6
4230 PRINT #0,"----"
4240 INK #0,4
4250 FOR i = 1 TO 3
4260   INK #i , 6
4270   POINT#i, Pt_in_3D(h(i)),Pt_in_3D(v(i))
4280   LINE TO #i, Ft_of_Pt2Line(h(i)), Ft_of_Pt2Line(v(i))
4290 END FOR i
4300 PAUSE
4310 :
4320 END DEFine views_in_3D
4330 REMark end of listing 3Views_added
```

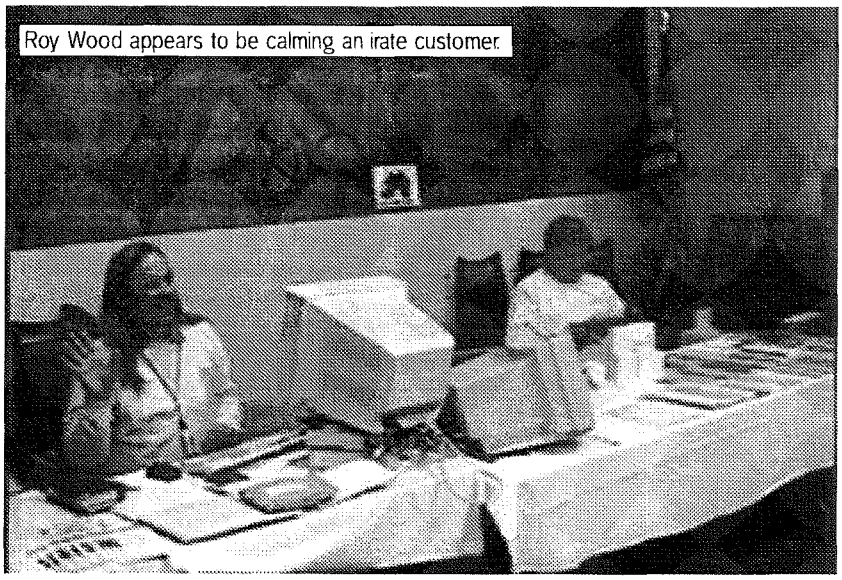
Quanta/NESQLUG US QL Show

by Al and Dorothy Boehm

Just driving there was a pleasure since White River Junction is located in one of the most scenic areas in America On Friday night we gathered at Than Wheelers, a restaurant next door to the Hotel Coolidge, the venue for the show. We had a fine supper, greeted old friends, and exchanged tales of the QL. The old hotel had considerable charm by which I mean the shower got too hot when someone flushed the toilet. Even so, the staff were quite courteous and I (Al) got a goodnights sleep. Dorothy on the other hand tossed and turned because it was too cold. I had to point out that it was May, winter was over, and the steam heat was turned off. Three years living in Alabama has spoiled her.



Jochen Merz and Dietrich Buder discuss QL events. I don't know what they said; it was in German!



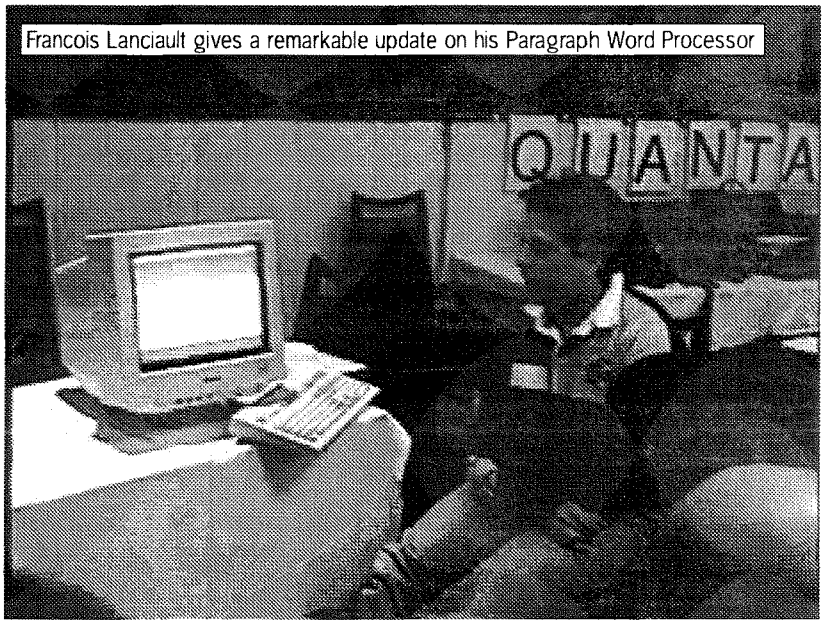
Roy Wood appears to be calming an irate customer.

Roy Brereton set up a Quanta table and renewed and started several Quanta memberships. He not only talked Bill McKelvey into joining Quanta, he got Bill to obtain sets of past years Quanta magazines. Tony Firshman came over to look at my Pandora. He pointed out several design flaws and flipped wires, changed my boot to start faster, and added a pilot light which lights when the RomDisq is still writing. He also gave me strict orders to wedge a piece of wood in it to prevent

The show was funded by Quanta who paid the rent of the show room. It was sponsored by NESQLUG which meant that Bill Cable, NESQLUG Director and Show coordinator did nearly all the work. The show was in a nice sized room with plenty of tables and electric outlets.

QLers watching one several demos.

the boards from wobbling. My Pandora is in much better shape now. I think. I was considering bringing the Pandora with me to QL2000 but now I won't since if that board isn't in just right, I will get a scolding! Seriously, thanks Tony. The MIDI Player played a variety of songs including the William Tell Overture and other classical songs as well as some swing, Elvis, and modern songs.
Herb Schaaf checked his MIDI cable to see if he had wired it right. He had. Zeljko Nastasic asked a lot of technical timing questions. Which I think I was able to answer correctly. I was able to give him a lot of good practical advice for MIDI on the Goldfire. Jochen Merz provided me with the color drivers for the QXL. I haven't had a chance to try them yet. My QD 98 was upgraded so it can call FiFi to find files. He also upgraded SMSQ/E for the SGC. But when I tried it on my Pandora with Aurora it did not load. ***[There seem to be problems only when a Qubide is connected - Tony Tebby investigates - Jochen]***



Francois Lanciault gives a remarkable update on his Paragraph Word Processor

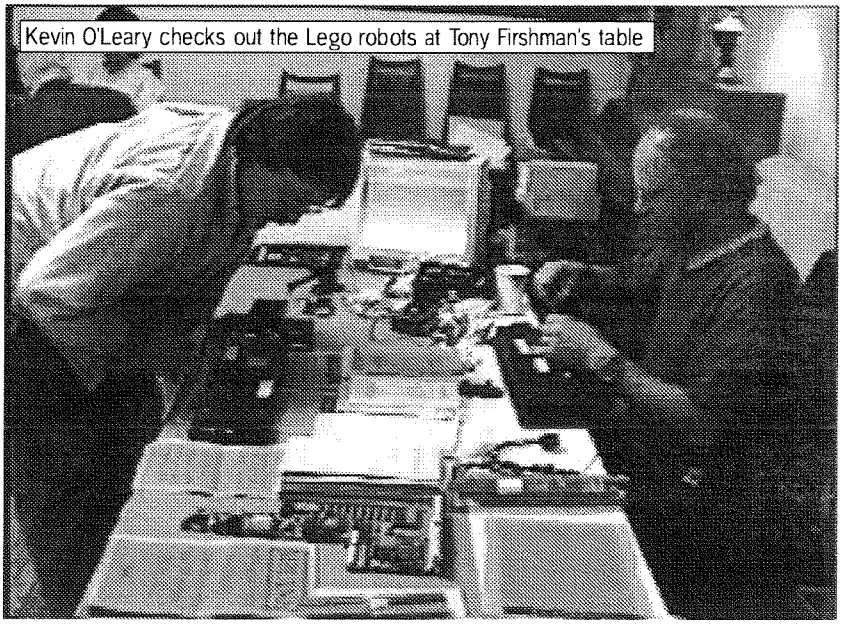
Al Boehm in front of his MIDI keyboard

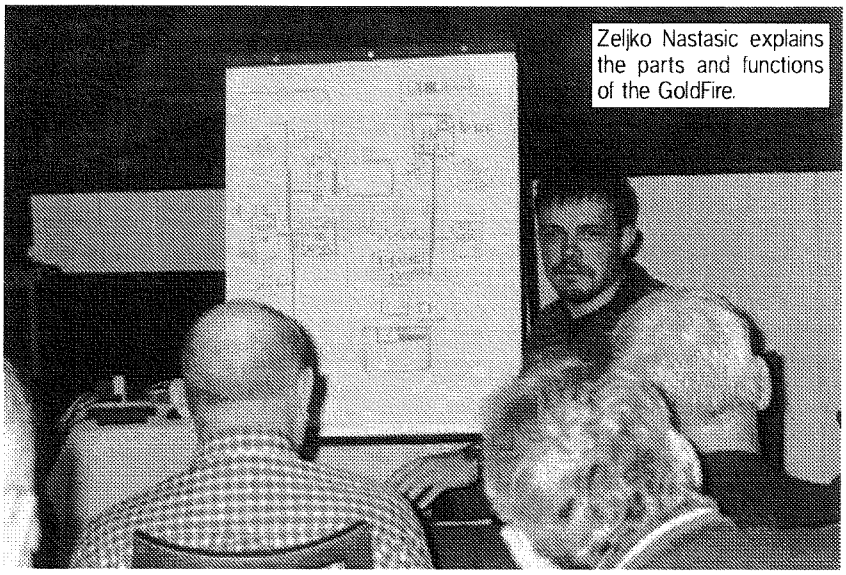
Back to the drawing board. Zeljko Nastasic gave an extended talk on the Goldfire. Many of the design trade offs were detailed. He explained the timing considerations in going from a fast Coldfire CPU to the slower QL peripherals. He also pointed out the need for going from a 32 bit bus to a 8 bit bus. A lot of work for his "black box" to do. He lamented the changes in chips which have slowed the project. All in all, a very thorough presentation. Of course, the sticky question was: when will it be ready? After so many delays this becomes a psychological barrier. Nevertheless, he bravely predicted that a prototype based on the revised standards would be ready this summer. Since he now lives in North Carolina, I suppose I could drive over and pester him. And I would too, but I know that would only make matters worst.
Francois Lanciault demonstrated Paragraph, the WYSIWYG word processorfor the QL. It requires ProWesS but now that ProWesS is freeware, cost

Roy Brereton representing ... QUANTA?

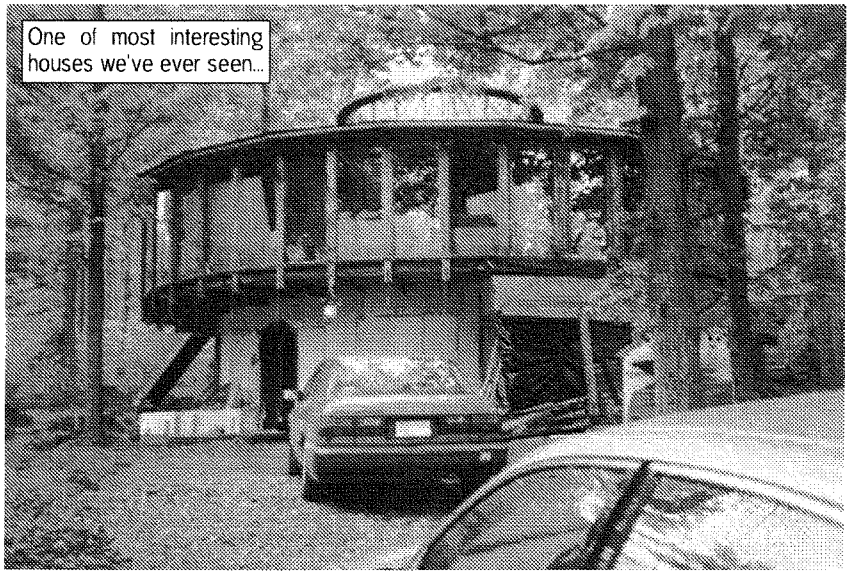
is no longer a big deterrent. A lot of the things he had not completed when he gave his demo at the Bedford, PA show two years ago are now working. Their were a lot of ooos and aaahs as the audience saw what it could do. It has some features that are more advanced than the top line very expensive word processors for the PC. The disadvantage ist hat the scalable vector fonts in ProWesS are relatively slow. Francois recommended a SuperGoldCard or better.

Roy Wood manned the Qbranch table. I had asked him to bring a few things with him for me. But I was so busy I forgot to buy them! Sorry Roy, I'll see you at QL2000 in Portsmouth.

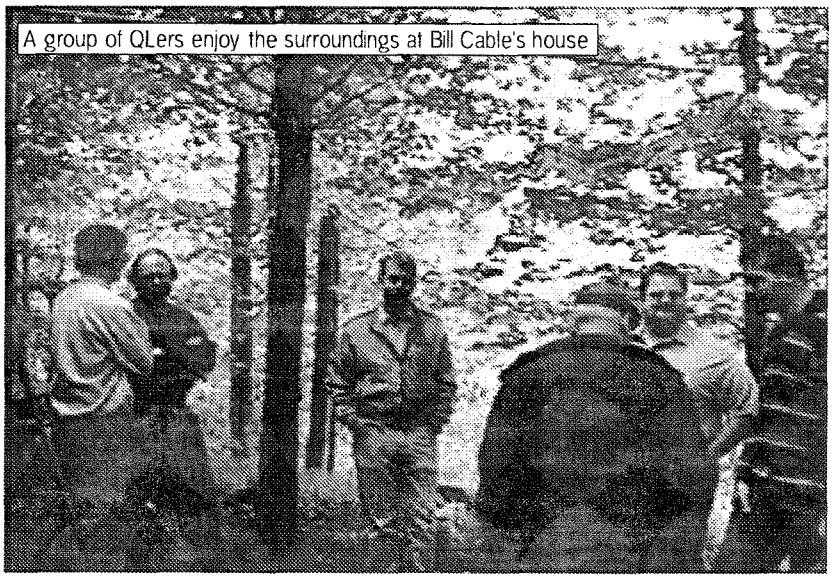

Kevin O'Leary checks out the Lego robots at Tony Firshman's table


Zeljko Nastasic explains the parts and functions of the GoldFire.

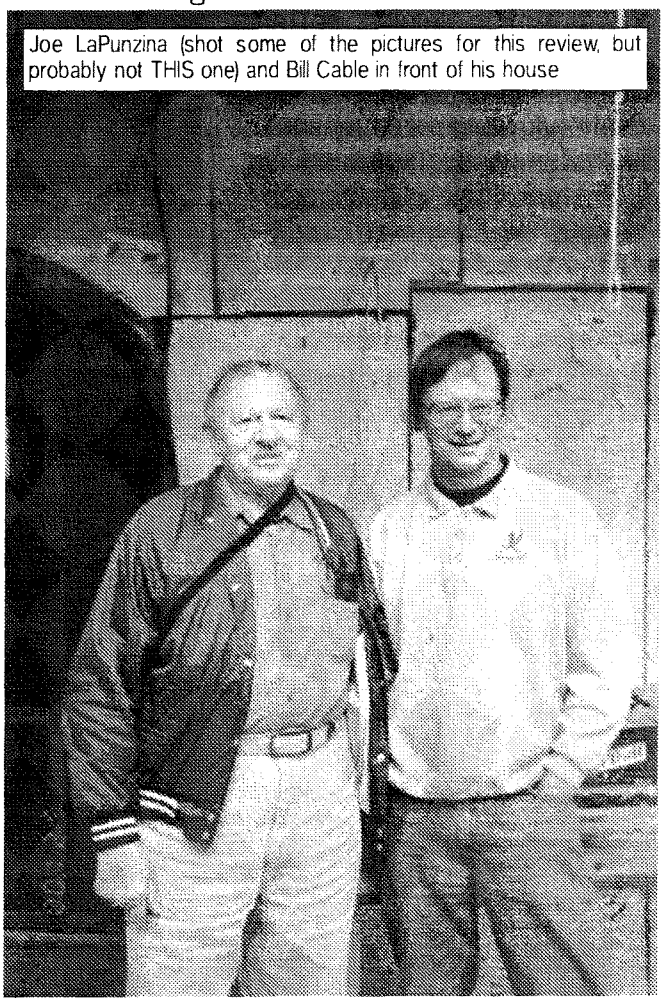
I ran a virtual table for George T. Morris' Bible Aids for the QL. In particular I offered the complete set of the aids on a Syquest Disk formatted for the QXL or the Qubide. I could probably also put the set on Zip or Syquest Disks for the Q-emuLator on the Mac. A couple of people were interested in the Bible Aids but they didn't have Syquest drives! In the meantime some people, mostly ladies, but at least one gentleman and also some kids, went on tours of the local area. Mary Boyle was the tour guide and Dorothy Boehm was the driver. We saw the Grand Canyon of Vermont in Quechee . The view was quite pretty and we looked at the gorge on both sides of the bridge. We ate lunch in a nice restaurant in Woodstock, and looked at the different shops and art galleries. The whole afternoon was very enjoyable for us all. After the show, we had a buffet dinner in the next room. The food was good. Bill Cable's wife Mary ran a contest to see who had been using the QL the longest. Al Boehm, Liz and Ian Padraza then sang a invigorating rendition of the QL Fight Song. We ended the evening by chatting in small groups. The next morning we prepared to move our venue to Bill Cable's house. This house


One of most interesting houses we've ever seen...

has to be seen to be believed. Bill designed and built it himself. It is circular. The ground floor is all one room. The second floor has individual rooms in sectors around the a central area which has a large glass coffee table which lets light from the observation deck above to filter down to the ground floor. Electricity is supplied by solar panels, a wind mill, and a back-up diesel generator in a shed. Tony Firshman was intrigued by the diesel engine. Or maybe it was the collection of old motorcycles that Bill has in the same shed. Heat is supplied by a wood stove. That is why Bill named his software company Wood and Wind. Water is provided by a hand pump and a rain water collection system. Roy Brereton ironed out plans for QL2000 with Al and Dorothy Boehm. Dorothy is organizing some activities for the computer widows and kids that come to Portsmouth. If you have never brought your wife to a QL show, you may want to bring her to this one. The wives plan to cluster at the Horizon Center at 10 AM on Saturday. They will organize a plan to tour and shop all of Portsmouth in just two short days. After the meeting at Bill's house, Dorothy and I drove Tony Firshman and Roy Brereton to Boston. Tony's flight required him to check in at 7:30PM. I told him I would get him there in time. And I did. That is, we got to the entrance of the airport at 7:30. The roads on theairport were one big traffic jam. We inched along. Finally about 500 yards short of Tony's terminal, we came to a complete standstill. It was now 7:40. Tony said not to worry, jumped out of the car, and got his luggage which included his folding bicycle. He put the bicycle together, put his luggage on theback, and away he went past the blocked cars. He made it OK. But our problems were not over. Roy needed to find a reasonable priced hotel. Everything in Boston was over $100. I told him I knew some good motels. We got to the first one. All filled up. We went to another one. All filled up. And another one. Same story. It seems this was the weekend that most colleges in Boston had graduation. Since there are something like 20,000 students in the Boston area graduating each Spring, hotel rooms are hard to find. After much retracing our tracks, it was after midnight when we finally found a vacancy at a run down motel. Well at least the price was right. Dorothy and I rested a day before driving over 1000 miles back to our home in Alabama. Francois Lanciallt has tentatively offered to host next year show. If he does, the US show won't be in the US at all; it will be Quebec!



A group of QLers enjoy the surroundings at Bill Cable's house



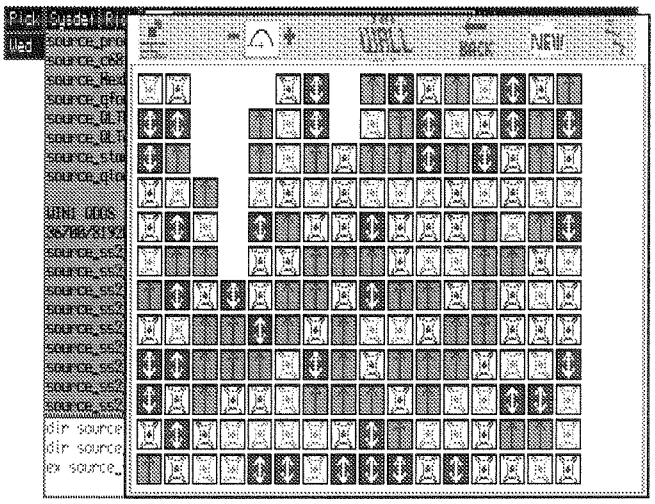
Joe LaPunzina (shot some of the pictures for this review, but probably not THIS one) and Bill Cable in front of his house

The Wall - a game reviewed

Norman Dunbar

I haven't reviewed any software for ages, too busy either at work or writing articles for the tutorial series on QL Assembly Language programming, and apart from all that, no-one has asked me!

Dilwyn asked me to review the new game from Jochen Merz Software, The Wall, which was written by Wolgang Lenerz, because the original reviewer had had some hardware problems and couldn't do it.



I have to say at this point that I don't do many games, Scrabble is about my limit - and I'm not very good at that either. This game has to be the most addictive I have yet come across and I have been playing it as often as I can.

From the name, you could be forgiven for thinking it was just another 'Breakout' clone. It isn't, it is more like a cross between Tetris and Breakout - but no bouncing balls in sight anywhere - just a wall (cue Pink Floyd music !)

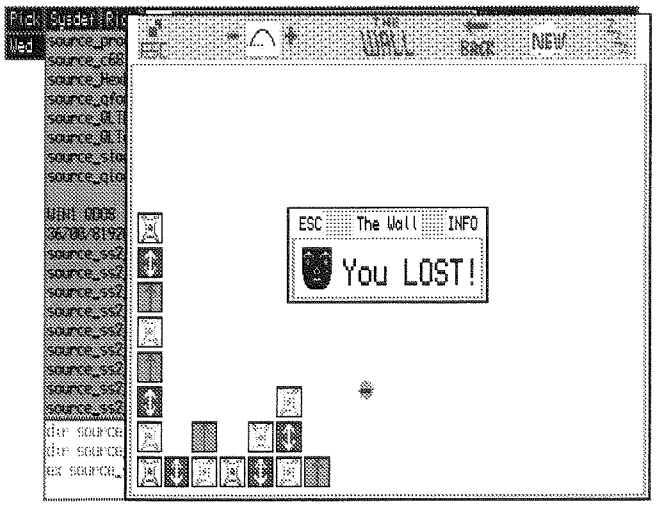
The game is so simple that almost no instructions are supplied. There is a manual, but it is brief - three and a half lines to explain the objective of the game, an additional 2 and a half to explain the keys used to control the game and one (almost) to explain how to make it more (or less) difficult. Nothing more is required.

The disc comes with four files on it, PTR_GEN and WMAN make up the Pointer Environment and WALL_EXE is the game itself. There is a BOOT file to load it all in. Minimalist or what?

I must confess to having a few problems with getting the game to work. My system is QPC 2 (Nice one Marcel !) and I have an older version of the PE loaded by my boot file. I simply tried to EX WALL_EXE but while the main game screen did appear, I got the NO ENTRY pointer and couldn't switch to the game.

When I ran the boot file on the disc, all was well. I don't think it runs very well with older versions of the PE, but, as a newer version is supplied - that is not a major problem. I have heard from Dilwyn that there are a few other possible situations where the game plays, but locks up and these are possible bugs that must be sorted out.

[They have been sorted out - I believe it had to do with the machine code BEEP. Now it works on all systems it previously failed on, mainly Aurora systems as far as I am aware. - Jochen]



Right then, what is the object of the game? Well, the object is to remove all the bricks from the wall by clicking on them with the mouse (or pressing space/enter if no mouse is used.)

When a brick is HIT or DO'd (?) one of two actions take place:

If the brick is in contact with other bricks, horizontally or vertically only - not diagonally, of the same colour or pattern then all the bricks of that colour or pattern are removed from the wall. The spaces left are filled in by bricks from above, or bricks from the right so the pattern changes.

If not, a small 'burping' noise is sounded from the speaker and no bricks are removed.

This continues until you have cleared all the bricks from the screen (unlikely!) or until there are no more bricks left that can be removed (most likely!).

One of the screenshots scattered about this review show the game in its early stages and a couple of screenshots show the usual (for me) outcome of the game.

JOCHEN MERZ SOFTWARE

Im stillen Winkel 12 D-47169 Duisburg
Tel. 0203 502011 Fax 0203 502012
http://www.j-m-s.com/smsq/index.htm

SMSQ/E Version 2.98 Update

Yes, it exists! The colour drivers not only work on the Q40, but also on the QXL card. 65536 colours are possible. The QXL interface has been improved too, the keyboard driver has been rationalised, and the speed of the serial ports and the parallel port increased. Depending on your PC, up to 115kBaud should be possible.

This Version is available for the other systems too. They cannot handle more colour, but "understand" the new colour commands and convert them to 4 or 8 colours. You can also define "desktop" background colour and image, even on the ATARI- and GoldCard-Versions.

Prices for Update/Upgrade

for ATARI ST, STE, TT	free
... with additional pages for the manual	DM 16,-
for GoldCard/SuperGoldCard	free
... with additional pages for the manual	DM 16,-
for QXL with add. pages for manual	DM 79,90

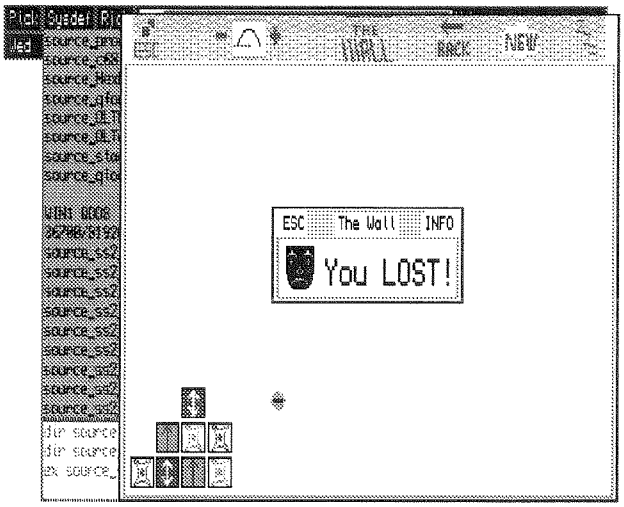
The new colour drivers for QPC2 will be coming soon!

TERMS OF PAYMENT

Postage and package **[Germany]** DM 8,99 (if total value of goods is up to DM 50,- then only DM 5,99). **[Europe]** DM 14,50 (if total value of goods is up to DM 50,– then only DM 9,50). **[Overseas]** between DM 14,50 (1 item) and DM 35,– (maximum). All prices incl. 16% V.A.T. (can be deducted for orders from non-EU-countries). E&OE. Cheques in DM, EURO, Eurocheques and Credit Cards accepted.

My best so far is only 3 bricks left. Every game is different and so far, I have not finished it. I keep trying though!

If you get to the end of a game, you can undo all moves made right back to the start and try another strategy. I have tried clearing from the top - no joy, from the middle - no joy, from the bottom - no joy, from the left - no joy and from the right - no joy (I see a pattern developing here!) and every combination of the above that I could think of - guess what? No joy!
All of that was on the easiest level which only has four different brick types. There is an intermediate level with 5 bricks and a hard level with 6 bricks - still no joy!



I find it hard to pick faults with the game - even though I have never won - because it has the qualities that a good game should have - it is fun, it is not too easy and you keep wanting to have just one more try - it is addictive. The other quality it has, is that it forces you to think about what happens if you remove such and such a row of bricks - sometimes the results can be surprising.
All in all, this game is highly recommended, but check if you might have problems with compatability first.
[I have solved it twice so far, but only on the easiest level ... out of 400 or 500 games - Jochen]

UNPICking LineDesign

Geoff Wicks

You will probably not have noticed it, but the last issue of QL Today contained a minor publishing revolution. If you look carefully at the Just Words! advertisement you will see the print quality is sharper than in most other adverts. It was the first non-paper Just Words! advertisement.
Strictly speaking this is not true. There was a paper copy, but a dodgy cartridge in my printer meant the print quality was too low for QL Today. As it was getting dangerously near the copy deadline, I asked Jochen how he would like electronic copy. He replied in one word, "GIF".
Usually I prepare my adverts in LineDesign, and my first instinct was to cheat by using a PC. I soon discovered this was not possible. I have DTP programs, but these cannot generate GIF files, and I have graphics programs that can produce GIF files, but which cannot handle paragraph text. There is no PC equivalent of LineDesign.
LineDesign has a GIF printer driver, but it does not work. Roy Wood suggested I should use instead the LineDesign PIC driver and then convert to GIF, but warned me that did not work either. Being a obstinate, bloody minded person, I decided to ignore his advice and make it work.
Creating a _pic file, the high resolution screen/pointer environment equivalent of a _scr image, from LineDesign is simple. You select the monochrome_pic printer driver and save the file to disk. Converting the PIC file to GIF was more complicated, particularly as I had to find a program to do it first.
When I do some QL research, I usually start with Dilwyn Jones' web site. It is the site I recommend to beginners or lapsed QL-ers as it is simple, but comprehensive. From there I go to Thierry Godefroy's site. This is generally regarded as the definitive QL site, although you have to be an experienced QL-er to get the best out of it. Neither site had the program I was looking for so then I tried Jonathan Hudson's. Here I discovered the program I was wanting, UNPIC (or QUNPIC the QL version).
When I confess to approaching Jonathan Hudson's site with some trepidation, I intend no disrespect to Jonathan. It is just that he has a thoroughness in file and documentation provision that can swamp we lesser programmers. Even zipped unpic is an intimidating 678K. Unzipped is over 4,000K. Do not be put off by this. The expanded files are usefully divided into subdirectories, which makes it easier to find your way around. In fact all you need to run the program on a QL are two files, qunpic, and a short, but complete, manual. Total length only 130K.

Using qunpic is simplicity itself. There is just a simple command line:

```
ex qunpic;"-v -g pic_file
gif_file"
```

-v is optional and switches on a counter to indicate conversion progress.
-g indicates you want to convert to a gif file. Other codes allow conversion to BMP, PNG, Postscript, TIFF and PCX.
Transferring a LineDesign page using qunpic creates a GIF image of 2376 x 3368 pixels, which is slightly larger than the QL Today page size of 2008 x 3070 pixels. It also produces an inverted image of the original page, and some final tidying up of the image was necessary. Once again I discovered shortcomings in PC graphics programs. Inverting the image was easy, but no windows program I possess is able to crop a GIF image of this size. I had to go back to an old MS-DOS program, Graphics Workshop, to crop the image to the required size.

In summary, if you have to send work to a printer, you can use your QL and LineDesign to prepare your art work and then send it electronically. However, a word of warning. Be sure to agree with your printer the graphics formats he can handle and the exact size of the graphic required so that no resizing is necessary. As Roy Wood reminded us resizing a GIF file containing large amounts of text and grey scales can produce an ugly distortion of the image.

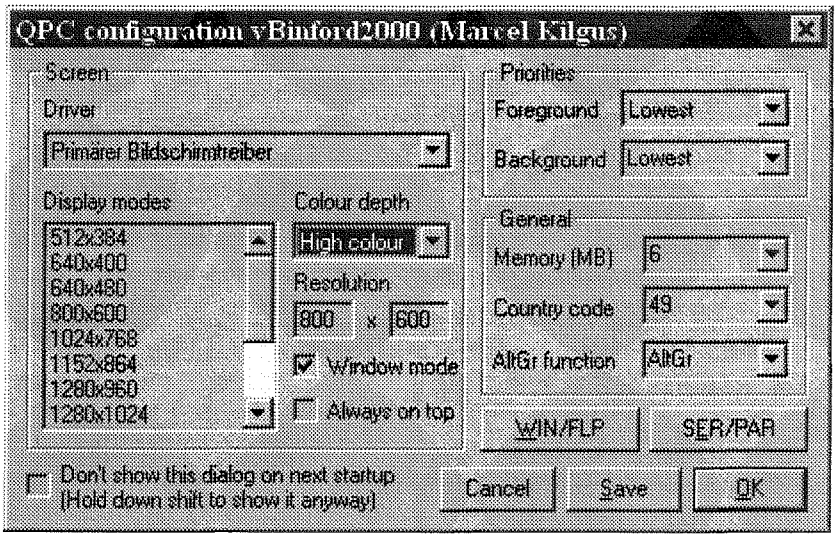
QPC2 ... the next Generation

Marcel Kilgus

Jochen asked me to write a few lines about QPC II v2.00 because he's a bit short of contributions. Well, I don't really like writing articles (a fact he knows, so he must be really desperate to ask me), but nevertheless I did my share. Now it's again YOUR turn. :-)

Why does it take so long?

Some people may have wondered why it obviously takes ages to implement the new colour drivers into QPC. Well, the real reason is that I got addicted to Terry Pratchett's Discworld novels, I devoured 6 of them during the last 3 or 4 weeks (of course the English originals which even take slightly more time to read than the German translations would). The second, more subtle reason is that it wasn't just a "throw that code in, assemble it and that's it" job, there was a lot more to do. Recently the problem with updating SMSQ/E for QPC was that I got loads of new sources with huge internal changes which, lacking any documentation at all, I couldn't quite follow. Previously I always used the QXL implementation as some sort of reference because there the conditions were similar: it has no own hardware and just needed code that passes the necessary bits and pieces to the PC software. But there were no new QXL versions, only the other platforms got updated, getting more and more incompatible to QPC's version. Finally I reached a point where I trashed all changes which were only internal ones and you, the users, wouldn't notice anyway (concerned mostly the DV3 driver) and waited for the big upgrade, the colour drivers, which again were supposed to include a working QXL version. Some time back in April they finally arrived, 6 zip files with a combat weight of 2,5MB. The first thing I did was to format a new partition and unzip the files there to re-implement the QPC version almost completely from scratch. It took some days to get to know the new QXL software, which had changed thoroughly, and some more days to understand the new GD2 driver. But after that the DV3 drivers turned out to be quickly adaptable and the screen driver worked after some days, too. However, even as I'm writing these lines SER and PAR are still not functional at all because I haven't come around to implementing them again. The SMSQ/E drivers changed hea-



vily and thus I'll have to rewrite most parts of my old code, which isn't really a bad thing because I suspect it'll improve this way.

What about windows?

In the past many users asked me whether I could implement a window mode to QPC so that it resides like all the other applications on the desktop rather than in its own dedicated full-screen window. I've always refused these requests for two reasons: I wasn't sure at all how it could be implemented with good performance and I wanted to wait for the colour drivers, because with their appearance I'd have to change the whole stuff again. Most people don't imagine how much trouble it in principle is to write a desktop based application which does not use any "high-level" graphics functions like "draw line", "clear block" etc. but has to have random control over every pixel in its window at any time. There are many things one has to keep in mind in order to write such an application: one doesn't have control over the colour depth, the resolution can change any second, the window can get partly hidden by another window and so on. Nevertheless I started working it out and I'm quite pleased by the result. QPC uses the so called "back buffering" technique, which means that it draws all changes into an invisible buffer and copies that periodically to the screen. I feared this would significantly reduce performance but the blitters (a special function in the graphic chips which are specialised in copying blocks of memory quickly and without help of the CPU) are better than I expected, the difference is clearly measurable but hardly noticeable. However, there's one requirement: the graphics card should have enough memory for both the visible screen and the back buffer otherwise the blitter can not access the data fast enough. Well, it would work despite the lack of memory, but the copying process gets really slow, which isn't a too big problem when using 512x256 or a similar small resolutions but I won't try 800x600 this way. This limitation may not be true for AGP system (= advanced graphics port, a special port for graphics cards that can give direct access to system memory), I'm not completely sure. I bought my first AGP card only two weeks ago and it seems to work properly with it but this may also depend on the graphics drivers. Modern blitters can scale up an image while they copy it so I left the window border changeable. This way you can sort of zoom into the SMSQ/E display. Trying this with a graphics card that doesn't provide the function in hardware is not a good idea.

The high colour mode is implemented compatible to the QXL (mode 32). Of course this mode is in principle much slower than the 4 or 8 colour modes as SMSQ/E has to process 8 times more data. Even the up to 20% faster emulation (I finally managed to debug the faster kernel I talked about years ago) can't compensate for this effect. However, I've implemented some sort of hardware-acceleration for some basic graphics functions resulting in windows that are painted and moved even faster than in the low colour modes. It's the same trick as Windows does since the "Windows acceleration graphics cards" were introduced about 10 years ago. Ever tried Windows with a graphics driver that doesn't support accelerated functions? It's horrible in comparison, even on up-to-date machines.

What else is there?

Most of the floating point routines within SMSQ/E will be replaced by PC code that uses the mathematical co-processor of the PC to compute the data. Graphics functions, SBasic and even compiled basic programs rely on these routines and experience a good speed boost (e.g. the test909-maths bench runs about 400% faster than before but that's of course a very extreme example).

The AltGr key is now supported natively but you can still configure it to act as Alt or Ctrl.

There are some more, mostly minor changes I don't list here so look them up in the new version2.txt file or at **http://www.deuschle.de/qpc/versions.htm** once v2 is available.

What about PC hard-disc access?

One of the most frequently asked question is if and when I'll implement direct access to PC hard drives. I've written most of the code months ago and it works more or less, though read-only so far, but there are still some problems that need to be solved. Development is put on halt while I'm working on v2, perhaps I find some time afterwards. It's just that I really hate writing device drivers.

So when will V2 be finished?

I hope to have it ready for the Eindhoven meeting in August but I don't guarantee anything. It all depends on my time, my mood, the weather (the latter two being somehow linked together) and how fast Amazon can deliver more Pratchett books.

You and Your Software - Just good Friends

Part 9 - Professional Blunders

Geoff Wicks

It may seen fanciful to suggest a computer program could make you physically ill, but read the following:

"What about the human cost, in suffering, stress, tears, and the many sleepless nights..."

The woman who wrote this gave up her job because of stress from the program she was using daily. An independent report called the software "illogical, inflexible and unforgiving" with "an unacceptably high risk of stress to staff". Eventually it was scrapped at an estimated cost to the British taxpayer of between six and nine million pounds.

The software had the unfortunate acronym of CRAMS - Case Recording and Management System - and was meant to revolutionise record keeping in the British probation service. In practice workers soon found it was incompatible with earlier data bases, and data entry a tricky process in which mistakes were easily made. In one case it recommended a sex offender as suitable for a job from which he was legally excluded.

We QL amateurs are sometimes apologetic about the simplicity and limitations of our programs, but the shortcomings of the professionals are much more serious. We can learn from their mistakes.

- How compatible are your programs with other software?
- Does your data handling program import from and export to the Psion programs?
- Does it allow transfer to and from other computer systems?
- How easy is data entry?
- Can mistakes be easily corrected?
- Do you provide information to allow other programmers to write mutually compatible software?

The incompatibilities between Perfection and QTYP and the lack of information about Text87 file formatting, have hindered me in the development of QL software.

Over the last year I have worked semi-freelance as a language specialist for several market research companies. It has been fascinating to see their different approach to common software problems. One company would win the "Golden Raspberry" award with ease. (I should perhaps add that raspberry is used here in its colloquial English sense of a loud disapproving noise.)

What do you make of the following commands on the same screen?

"(ENTER 'DK' TO EXIT SECTION BUT CONTINUE SCREENING)"

"Reply may not be NULL or DK or REF"

"Reply may be one of the above"

I think I know what is intended by these instructions. The first is meant for supervisors and programmers only. The other two are for interviewers and follow a list of menu items, but the statements have been placed in the wrong order. Put them the other way round and they make sense.

Writing meaningful instructions on a screen requires considerable skill. You have to get your message over with the minimum of words. To see how the experts do this switch on the subtitling for deaf people on your television set, or study the teletext news reports. When writing screen instructions, concentrate on the new user doing only simple things. The advanced user will quickly learn for themselves, or will look things up in the manual.

Your screen should make a clear distinction between instructions, menu commands and other items. To do this you could use windows for different types of information; colours to emphasise parts of the text; or a different font size or font type.

In market research the contents of a screen have to be quickly assimilated by a telephone interviewer. They include the question to be asked, the permitted answers and other instructions. For example, there is often an instruction that "Don't know" or "Refusal" are permitted answers, but the respondent may not be told this. If the instructions are not clear, the interviewer stumbles over the interview and comes over as unskilled.

Not all market research companies have colour monitors, and some have to work with combinations of inverted and bright text to separate the different types of screen information. The "Golden Raspberry" company is inconsistent in its use of these, and relies on the interviewers knowing the script. More seriously, if some monitors are not in the correct mode, an interviewer in the middle of an interview can

discover some of the questions are not visible on his screen.
A common market research question is brand recognition, where the respondent is asked to name the companies he can think of. There can be as many as 30 company names spread over three screens.
An interviewer has to be able to flip quickly backwards and forwards between these screens. One company uses function keys to do this, which has the advantage of a single key press, but is difficult to remember. The "Golden Raspberry" company uses "+" for next page and "-" for the previous page, a poor choice as the former requires the shift control and thus the use of two hands. As the company does not provide head sets, interviewers have to hold the telephone between their shoulders and their ears while carrying out keyboard operations.
I suggested Page Up and Page Down would be better, but then discovered some non-standard keyboards without these keys. Some have the control key where the shift key usually is. If interviewers are not careful, they can press the wrong key and crash the computer.

A Dutch project had as one of the companies "PTT Telecom". Dutch interviewers protested the name had been changed to "KPN" years ago, but were told the script was holy and could not be changed. A few months later the survey was rerun with a new team, and although the name had been changed, it was kept in the same place so the list was no longer in alphabetical order. Who says poorly written software cannot cause stress?

What are the lessons for the QL? Mainly to think constantly of your users, whose knowledge and work patterns may be different from yours. Look at every operation involving the keyboard and assess if this is the simplest way of doing the task. Your users will not be making phone interviews, but they could be making notes or could suffer from a visual or physical handicap. Most users will not be using SMSQ-E on a Q40. Some will have slow, older systems or even a low resolution monochrome monitor.
One final example of user friendliness and unfriendliness. Quill and Text87 allow you to view a directory list, but neither list is in alphabetical order. Compare this with a Menu Extension (part of QMenu) list (e.g. file_select$), which is not only in alphabetical order, but can also be filtered by extension. Reviewers have sometimes criticised Just Words! programs for not providing a directory command. The reason is that it would add to the length and complexity of the program. The Menu Extension does the job far better than anything I could write. The Menu Extension and the Pointer Environment extensions are now available on one disk for only £5. A must for every QL user.

Next time: Commercial or non-commercial?

WXQT2 Review

by Dilwyn Jones

Shock, horror! QL Today reviews a Windows/Linux program in its pages.
Without shame, I am going to review what I consider to be a gem of a program from Jonathan Hudson.
Last year, I made a lot of use of another utility from Jonathan Hudson, called WXQLtools, which allowed me to transfer files between the native hard drive on my PC and QL floppy disks. There was also a Linux version. WXQLtools acted as a front end on the QLtools utility, which allowed transfer of files between those media.
I was also aware of a program QXLtool from Jonathan, a rather unfriendly (to someone like me who was unfamiliar with Linux and DOS-like commands and found the commands hard to memorise) which allowed the transfer of files from a QXL.WIN filing system used by QL emulators like QXL, QPC, uQLx and recent versions of QemuLator (PC version).
Late last year, a disk arrived in the post from Jonathan Hudson with a Beta copy of a program called wxqt2. The rather uninspiring name concealed what was to become an essential utility for my computing setup.
Essentially, this program is a pointer driven front end for the QLtools and QXLtool programs, which runs on the native operating system of the computer it is used on. It comes in both Windows and Linux flavours. I am not a Linux user (my brain has enough trouble coping with one non-QDOS operating system let alone two of them!) so my comments will be restricted to use of the Windows version on my Windows 95 desktop computer and my fiancee's laptop PC using an earlier version of Windows 95.
My "QL" setup is based on a Windows 95 desktop PC running QPC (it actually has both QPC1 and 2 installed, though only QPC2 gets regular use). The use of WXQT2 is in fact independent of the QL platform, as long as that QL plat-

form uses the QXL.WIN files favoured by Tony Tebby for SMSQ hard disk file storage.

I downloaded the latest version of wxqt2.zip from Jonathan's Web site recently and installed that on my hard disk. I had encountered and reported a few problems running it on my Windows 95 system, and Jonathan made a few changes to help prevent the problems I was having - it now seems to run without problem on both systems. So the comments from here on all refer to the latest version.

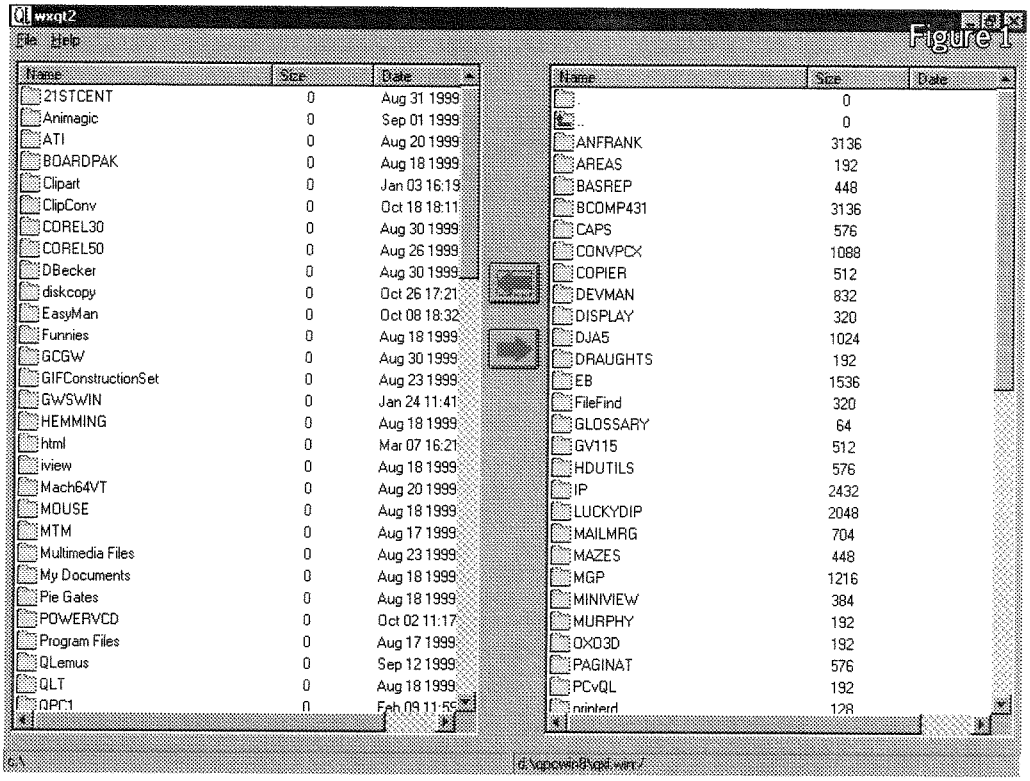


Figure 1

When I downloaded and unzipped these files on my PC, I found a large number of files and directories. As is his normal practice, Jonathan provides a mixture of source files, copious documentation and versions for Linux and Windows, and so this leaves you with the task of reading the various documents to find out exactly what you need for your particular system. In my case, I needed to place the following files in the same directory:

`QXLtool.exe` (Windows version)
`QLtools.exe` (DOS version)
`wxqt2.exe` (Windows version)

I then created a desktop shortcut by right clicking on WXQT2.EXE, selecting the Create Shortcut option and I was ready to go. Simply double clicking on the shortcut icon from Windows 95 started up WXQT2 for me.

Figure 1 shows the opening screen. This gives you two large windows, one showing a source drive, the other showing a destination drive. Either box may be defined as one of your QXL.WINs holding your QL files, either may be a floppy disk drive, or either may be a native Windows/DOS hard drive. You can define what is what by entering the File menu, selecting the Options command, which brings up the screen shown in Figure 2. Here, you can set which device is represented in which window.

- `wxqt2_zip` - wxqt2 front end files
- `qxltool_zip` - qxltool files (must be v1.13 or later)
- `qltools_zip` - QLtools files (must be v2.14 or later)

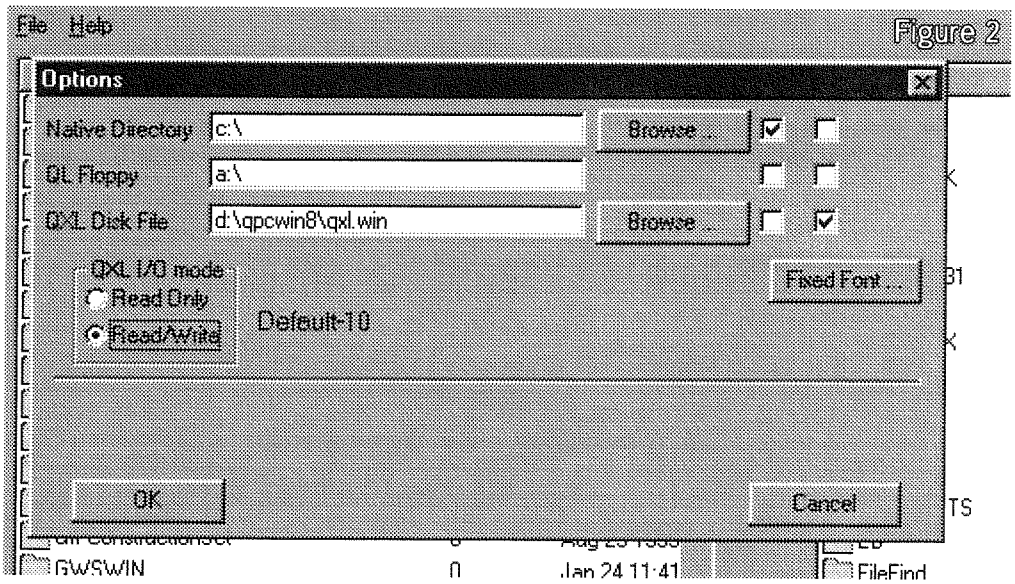


Figure 2

The left hand side can for example be set to the native hard drive of your computer, in my case drive C:
The right hand side can be set to be your QXL.WIN file (or floppy disk drive). In my case, I have several of these QXL.WIN defined (QPC allows you to define up to 8, whereas QXL users may be limited to one). So I entered C:\QXL.WIN here for my WIN1_ drive. I also have a couple of QXL.WIN files (my WIN7_ and WIN8_) defined as being in a sub-directory on my second hard disk, drive D:
Of course, I can only access one of these QXL.WIN files at a time in WXQT2. And I would not normally attempt to define both windows as QL drives since copying files between two QL drives is of course better done in the QL emulator itself!

You will notice from Figure 2 that QXL.WIN files can be set to be read only, or to allow WXQT2 to write to them as well. Jonathan complains in his documentation about the state of documentation available for QXL.WIN files, which he has had to partly work out for himself and partly from other people's notes, so if you feel unhappy about having his software write new files into your QXL.WIN because of this, he gives you the option of telling WXQT2 it is not to write to the QXL.WIN file. Your choice.

Another reason is that some QXL.WIN media simply cannot be written to - CD-ROMs for example. WXQT2 seems to be able to read files from QL CD-ROMs (QXL.WIN on ISO-9660 CD-ROMs I made for Q-Celt Computing) although I don't know if WXQT2 was meant to be used in this way - it seems to work.

I have been using the latest version of WXQT2 for some time now and it has yet to trash a QXL.WIN on my system, so I trust it to write to my QXL.WINs. So far anyway.

You can either type in the path and name of the QXL.WIN file (use the path name format of the operating system on which the program runs) or click on the Browse button to select the appropriate drive and file. You can also specify the fixed pitch font to be used for file-name displays. Once happy, click on the OK button, and there is a nerve wracking pause while the program reads the drives to see if it can achieve whatever you specified. Early versions of the program would often bomb out on my system with either the bog standard Windows 'illegal instruction' errors or lock up while attempting to read the drives. More recent versions seem to tidily tell you that you got something wrong and give you the chance to get your brain in gear before your Windows installation disappears to that great DOS-house in the sky.

One irritating feature I found under some versions of Windows was that if you attempted to go further up the directory tree than the root of the drive specified, e.g. if you have got to root of drive C: and click on the 'go back up one level' icon various problems could occur. The early versions in particular did not seem too comfortable with handling the root directories of drives in Windows. Attempts to copy files to or from the root directories were either ignored (copying appeared to take place, but the file did not in fact appear there when you went to find it later) or caused a total Windows crash - on more than one occasion I had to reinstall Windows 95 on the laptop after this. Recent versions have been much more secure in this respect. If I have understood correctly, there are two possible reasons for this - the early libraries for the compiler used to make the program and also (correct me if I'm wrong Linux users) it seems that Unix/Linux guys don't like the idea of placing user files in root directories. I suppose even if there is a specific reason for this, it is only good tidy practice not to litter your root directory with user files, even on a QL hard disk system.

The File menu also contains an option to create a new QXL.WIN file - you are asked to specify the path and filename, the volume label and the capacity (in MB).

The third option on the File menu is the standard Exit command to leave the program.

The only other command is Help. One of the options brings up a screen about WXQT2, while the other accesses the main Help system. The first time you use this, you may get the error message that you should set the environment variable WXQT2_HELP_CMD to be able to access the Help system. Read the instructions again!

To copy files between the two media, simply highlight the files to be copied. Click on the name of the first file, use CTRL left mouse button for second and subsequent names to add to the list (or draw a rubber banded rectangle around a group of filenames if preferred) then click the arrow between the two windows to send the

files to the drive shown in the other window. Copying proceeds fairly quickly and after a while the content of the windows are redrawn to represent the updated contents. If copying is likely to take some time you will get a progress warning window and message.
If you right mouse click on a window containing the list of files on a QL format drive (floppy or QXLWIN) you get a little QL popup window with a few commands for handling the files - see Figure 3. Here you can Delete, Format, Make directories, see the QLtools or QXLtool version numbers (depending on whether you are accessing floppy disk via QLtools or QXLWIN via QXLtool) and get an info report which tells you the volume label, drive capacity in sectors and so on. You can also View a file - this displays the content of a text file in a window with a scroll bar to allow you to scroll through the text.

At the top of the list of files there is a pair of icons, one containing a single dot and the other a double dot and an arrow icon pointing left and up - clicking on this takes you back up one directory level. Clicking on a directory name opens that directory and shows the list of files in it - rather like a DO on a directory name in the QPAC2 files menu. Clicking on the single dot item seems to do no more than refresh the content of that window.

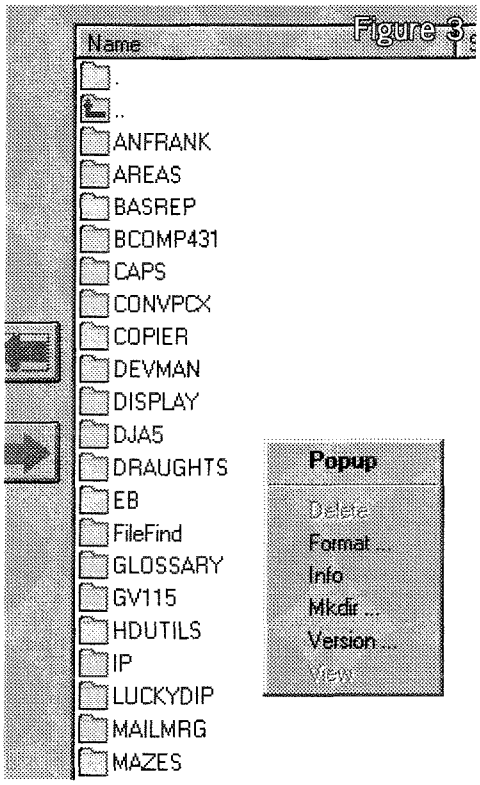


Figure 3

WXQT2 in common with many of Jonathan Hudson's programs is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation.

So there we have it - once you have waded through the documentation and sorted out what goes where and what is actually needed as a minimum set of files to run the program, this is easy to use and very effective in terms of what it does.

Some of us have been moaning for years about how long it took to copy files via floppy disk between the QL emulators and the computer's native format hard drive. WXQT2 solves that problem at a stroke. Until I can do all this from QPC2, my current way of working is most things done in SMSQ/E on QPC2 and dive into Windows 95 to achieve anything else such as internet and email access which I can't yet do from QPC2. It used ot be a real chore to transfer files between the QL and PC. And now it is easy, although the problem of incompatible file types remains and you have to resort to programs like Textidy to help with that problem. WXQT2 does not try to make Quill DOC files into Word files or anything clever like that, it simply moves files between otherwise incompatible media and does it in a way which is about as simple and straightforward as possible. Linux and Windows users will appreciate the fact that the same program is available for both platforms.
If you need to transfer files between QXLWIN, floppy disk and Windows or Linux media, get a copy of WXQT2, QLtools and QXLtool - hopefully you'll be as impressed with this software as I was! In my eyes, WXQT2 has joined software like Qascade from the same author which rank as "must haves" for me.
If you have internet access, you can download these programs from Jonathan Hudson's Web site:
http://www.hedwig.uklinux.net/index.html

Programming ProWesS in SBASIC - Part 2

Wolfgang Lenerz

In this second instalment of this (mini) series, we'll see how to activate Prowess objects. To recap, in the first part of this series, we saw that a Prowess window is made up of "objects", which are created with the PWcreate function. The first object to create is always the outline of the window, which is just a sort of container the purpose of which is to contain (or "own") all of the other objects (some of which are very powerful).
Once creation of the outline is done, you can

create other objects for, and contained by, it. The entire suite of objects is a system. Once you have created a system, you can activate it. Activation means that the window will be drawn on the screen, and Prowess then processes the key-strokes/mouse clicks.

Activation

Activation normally only concerns an outline object: unless specified otherwise in the documentation, you should never attempt to activate other objects - havoc may ensue. This actually seems quite logical, since, as was already mentioned, the outline object is the object that will contain all other objects in your system.

When you activate the outline with this keyword, control of your program is passed to Prowess. Prowess then (and only then) actually paints the window on the screen with all the objects you have PWcreated. Once the window is drawn, Prowess watches over what happens with the pointer within the window, and also whether the user presses any special keys etc... Just like in the Pointer Environment, the user can HIT or DO an object such as a menu item. There are, however, also some other special events (such as sleep, pointer in window, drag, and many more). As a general term, Prowess uses the word ***'indicate'*** to show that an object was in some way selected to do something. As we will see, an object is indicated if control should pass to this object, for example if the user hit or did this object.

Then generally whenever an object is indicated, (i.e. at some pre-defined point), control will come back from Prowess to your program and Prowess will tell your program why it came back, i.e. what has happened (such as the fact that the user clicked on a loose item). For those who know the Pointer Environment, this is not (much) different to the Read Pointer loop under QPTR.

Activation of the outline is achieved, not surprisingly, with the ***PWactivate*** keyword. The syntax of this keyword is as follows:

```
mempointer=PWactivate (object, mempointer,
object_hit,add_info,hit_or_do%)
```

The parameters to this function have the following meaning:

- ***object*** is the outline object to activate, as returned by the call to the PWcreate function.
- ***mempointer*** is the result returned, and is also used as a parameter in the call to PWactivate. It is a pointer to a special place in memory, but this is entirely managed by the Prowess SBasic Interface, so you don't really need to concern yourself with it. However, you must know that this parameter **MUST BE 0 THE FIRST TIME YOU USE PWactivate** for this object. After this, it is maintained by the system itself and you do not have to worry about it - actually you should never again tamper with this variable. You must just know that, on return from the PWactivate function, this pointer will be 0 when the user clicked on the QUIT item of your window, indicating that he/she doesn't want that window anymore. You should then remove the outline object and (possibly) stop the program.
- ***object_hit*** tells you what object was ***hit*** (or ***done*** or otherwise ***indicated*** by the user) and which thus caused Prowess to return control to your program. This enables you to make a SELECT on all objects which are likely to be actioned by the user, and thus direct the flow of your program to where you deal with this object.
- ***add_info*** is some additional information about the object that was hit or done or otherwise indicated. Exactly what this additional information is depends on (i) the object indicated, and (ii) the cause for the return from the ***PWactivate*** call. For some objects (and even more often than not), this parameter actually doesn't contain any useful information... The content of this variable -if any- will be explained later when we explain the different objects and their tags: if a certain tag for a certain object causes this variable to contain useful information, you will be told when this tag for this object is described. In most cases, you can just ignore this variable.
- ***hit_or_do%*** (an integer variable) contains information about whether the user ***hit*** your object, or ***did*** it, or in what way this object was indicated, since indeed, this might be of some importance. For example, you might want the program to react differently to a loose item being hit, or it being done. If the value of this variable is 0 upon return, then the object was ***hit***. If this variable is 1, the object was ***done.*** This variable can have many other values - the value depends directly on which action routine you indicated when setting up an object. This will be explained in much more detail below.

So, once you have created all of the objects, your program will be actually a simple loop, something like this:

(...definition of your objects here - you are presumed to have defined an object called outline

and another one called item1, which is a loose item...)

```
mem=0:object_hit=0:hit_do%=0:add_info=0
REPeat loop%
  mem=PWactivate(outline,mem,object_hit,
 add_info,hit_do%)
  SELect ON mem
     =0      : PWremove outline:STOP
             : rem user pressed QUIT
     =item1 : take_care_of_this_item
          (...)
  END SELect
END REPeat loop%
```

This is actually a very simple program structure - a loop which terminates when 0 is returned in ***mempointer***. Indeed, we now know that, when this parameter is returned as 0 from the PWactivate call, than the user pressed the QUIT item of the window, so he wants to quit this window. If this is the main window, that means that he wants to quit the program altogether, hence we remove the outline (PWremove, explained in more detail later) and STOP the program. If, on the other hand, the user indicated the item1 item, then we do whatever this item is supposed to do.

Defining objects to return

So, Prowess draws the window and then handles everything else, checking where the mouse is, whether an item in indicated and so on. This would mean that the ***PWactivate*** call would never return, so there must be a way to tell Prowess to relinquish control and return to SBasic. This happens automatically whenever the user quits the program - in that case, the ***PWactivate*** function returns 0, as we have seen.

All that needs to be explained now is how to tell Prowess to make a return from the ***PWactivate*** call for other reasons (i.e. indication of an object). This is achieved whenever you create (and, sometimes, change) an object. As was already explained, all objects have tags that modify the object or its behaviour in some way. For example, an outline object has a tag called ***"PW('outline_quit')"***. If you pass this tag when you create the outline, then the outline has a QUIT item. If you don't pass this tag when creating the outline, then the outline doesn't has a QUIT item.

If you now look at the tags for a determined type (you can find them in the manual for the Prowess

SBasic interface), you can see that some objects permit "do routines", "hit routines" or other ***action routine tags*** to be passed to an object. An example would be the tag PW("OUTLINE_ACTION_DO") for the outline type of object. If you use this tag, it must be followed by a routine which is called from the Prowess Event Handler. In other words, you must tell Prowess what routine to call when the user indicates the DO item in the window.

This means that Prowess itself, once it has drawn the window, is just a loop that checks what the user does with the mouse or the keyboard (producing an "event"). This piece of Prowess is the Prowess Event Handler. If the user indicates the DO item, the Prowess Event Handler checks whether there is a routine it should go to (the one indicated by you) and, if so, it jumps to that (machine code!) routine. That routine then does what it has to do (in this case, it probably returns control to your program), and then returns control back to the Prowess Event Handler.

Of course, for the SBasic programmer, the best thing would be if there was a way to tell Prowess to call a determined SBasic function or procedure directly upon such an event. Unfortunately,that isn't possible since you cannot call an SBasic procedure from within a machine code program. So there are several new keywords, which, when called from the Prowess Event Handler, will return control to SBasic: ***HIT_ROUTINE, DO_ROUTINE*** and many other action routines, which are explained elsewhere in the manual. You just use these as parameter whenever a tag requires an action routine. For the technically minded, the next paragraph is a slightly more complete explanation:

A fake return

Thus, whenever you use these two keywords to indicate that, upon an indication of an object, Prowess should come back to SBasic, this is indeed done: the ***PWactivate*** call returns with a certain value in ***mempointer*** (other than 0). You must be aware however, that this is a ***fake*** return! In fact, Prowess doesn't know that it has returned control back to SBasic. It still thinks that it is in the routine that was called from within its Event Handler, since these action routines return control to SBasic without exiting from the Prowess Event Handler. As we saw above, Sbasic programs under Prowess are just a loop wherein the PWactivate function is called repeatedly. On the first call to PWactivate, the mempointer parameter is 0 - this indicates to the SBasic interface that this is a true activation call. On return from the PWactivate call, when the return was caused by the action routine of an item, the mempointer parameter is not 0, and thus is not 0 when passed back to PWactivate in the next instance of the loop. This way, the Prowess SBasic interface knows that this is not a real PWactivate call, but actually the return from an action routine. It then returns to that action routine, instead of making a true activation of the object.

Anyway, it is thus very important that you always check on the ***mempointer*** variable returned by ***PWactivate***. If this is 0, then a clean return was made from Prowess - this can also be seen by the fact that the window drawn by Prowess has disappeared: whenever a clean return is made to SBasic, the window is undrawn first.

If the value returned by ***PWactivate*** is not 0, then a fake return was made, and you must loop back to this call and go back to Prowess so that a return is made from the routine called by the Prowess Event Handler. In that case, **DO NOT CHANGE THE VALUE OF THE *mempointer* VARIABLE!** This way of handling events also explains the loop structure of the above example: a simple loop, always calling upon ***PWactivate***, until 0 is returned. If you do not do this, but rather call the ***PWactivate*** call completely anew (i.e. with mempointer = 0) then you will certainly loose some memory (at least 4 KB) this is a guaranteed minimum). Apart from that, it should be safe to do so, but there is a risk that you might also confuse (i.e. crash) your copy of SBasic and/or Prowess and/or the entire system. So don't!

For reasons which will be more closely explained later on in this series, it is never a good idea that the outline and the mempointer variables be LOCal variables.

One last thing must be explained before we go on to the first example program, and that is the question of setting a window's outline, which is achieved with the PWoutln keyword. Please note that the outline of a window, and the Prowess outline object, are two very different things!

PWoutln: setting a window's outline

With PWoutln, you set the "outline" of a window. In an ideal world, this keyword shouldn't be necessary since the outline of the window is normally set automatically by Prowess whenever you activate the outline object. The keyword is necessary, however, when you use the SBasic interpreter.

You may remember that, in the introduction I mentioned that Prowess is a window manager, and still needs the Pointer Interface. The latter is responsible to let programs know that the pointer

is in their window, and what is happening with it. However, to be able to do this the Pointer Interface itself has to be ***truly aware*** of the window, which just means that it has to know a certain number of things about the window.
Setting the outline of a window tells the Pointer Interface these things about the window, and also tells the Pointer Interface that we want it to manage this window. Now, normally, this is done transparently by the ***PWactivate*** keyword when we activate the outline object, so we wouldn't need to worry about this unduly. However, one peculiarity of the Pointer Interface is that the outline of the ***very first window of a program*** must always be set in this manner - if it isn't, setting the outline of other windows in the program won't work correctly.
This is no problem when your program is compiled ***(with the 'WINDS' option turned OFF*** - we will come to compiling programs later on in this series. Nor is this a problem if you EXEC'ed an SBasic program so that it has no windows open at the start. In the interpreter however, windows#0, 1 and 2 are already open before you RUN your program - and the outline of window#0 is not set, window#0 being the very first window of the program! So, the ***PWoutln*** keyword is used to set the outline of window#0 (the first one to be opened for the interpreter) of the SBasic you're working in. After that, everything will work alright.
You can test this by using the test program below. REM out the part setting the outline of channel#0 and run the program. You will note that the windows are still drawn (more or less) correctly, but you cannot get the mouse to work... In other words, the Pointer Interface doesn't tell your program correctly that the pointer is in the window, unless you used ***PWoutln.***

The syntax of this keyword is as follows:
`PWoutln [#chan%],x_size%,y_size%,x_orig%,y_orig%`

and the parameters for this are:
- ***#chan%*** is the channel ID of the window whose outline is to be set. Normally this would be channel #0. You can leave this parameter out, it will then default to channel #0.
- ***x_size%*** and ***y_size%*** are the sizes you wish to give to your outline. ATTENTION: any window open for the program MUST fit within this outline: If you set the outline to a small size (say 50,50) you can't open a window bigger that 50,50, you will get an error out of range if you try!
- ***x_orig%*** and ***y_orig%*** are the x and y origins of (outline of) the window.

Please note that after you have used this command, you will have to redefine the windows for the job (i.e. channels #0 ,#1 and #2). The example programs use a procedure called "set_windows" just for that.

The first program

Enough theory, let's write our first Prowess program in SBasic. Here it is:

```
100 init
110 main
120 STOP
130 :
140 DEFine PROCedure init
150 REMark initialise/make the outline object
170   set_windows                  : REMark make windows ok
180   text$='Prowess Sbasic test 1'&CHR$(0)
210   outline=PWcreate(0,PW('TYPE_OUTLINE'),PW('OUTLINE_TITLE_TEXT'),text$,
                       PW('OUTLINE_QUIT'),PW('OUTLINE_QUIT_KEYPRESS'),27 )
220 END DEFine init
230 :
240 DEFine PROCedure main
245 REMark the program loop - wait until event happens
250 LOCal loop%,object,add_info,hit_do%
260   mempointer=0:object=0:add_info=0:event%=0
270   REPeat loop%
280     mempointer=PWactivate(outline,mempointer,object,add_info,event%)
290     IF NOT mempointer:EXIT loop%
300   END REPeat loop%
310   PWremove outline
320 END DEFine main
```

```
330 :
340 DEFine PROCedure set_windows
350 LOCal not_compiled,upper%,xo%,yo%,xs%,ys%,ysize_0%
360   not_compiled=IS_OPEN(#0)      : REMark window#0 open?
370   IF not_compiled
380     xs%=0:ys%=0:xo%=0:yo%=0
390     PWscrsize#0,xs%,ys%,xo%,yo%
400     upper%=28:ysize_0%=50
410     PWoutln#0,xs%,ys%-upper%,xo%,upper%+yo%
420     WINDOW#0,xs%,ysize_0%,xo%,ys%-ysize_0%
430     WINDOW#1,xs% DIV 2,ys%-upper%-ysize_0%,xo%+(xs% DIV 2),yo%+upper%
440     WINDOW#2,xs% DIV 2,ys%-upper%-ysize_0%,xo%,yo%+upper%
450     BORDER#1,1,255:BORDER#2,1,255
460     PAPER#1,2:PAPER#2,7
470     INK#1,7:INK#2,2
480     CLS%0:CLS%1:CLS%2
490   END IF
500 END DEFine set_windows
```

So what does this program do? The first two lines call the 'init' and 'main' procedures. Not surprisingly, the 'init' procedure initialises the outline, and the 'main' procedure contains the main program loop. After this, the program just stops (the STOP statement isn't strictly needed, I just put it there to show that this is indeed the end of the program).

The set_windows procedure

The ***set_windows*** procedure is the first thing that is called from the 'init' procedure. ***set_windows*** is a procedure which you could (should!) use in every program using the Prowess SBasic Interface. It makes sure that the outline of the window is set if we are not in a "compiled" program. This it is a procedure that will be used in very example program, I'll explain it at length:

As you can see, this checks whether window #0 is open (line 360). To do this, it uses the "IS_OPEN" function, which is a new keyword, supplied as part of the Prowess Sbasic Interface package. This function can tell you whether a channel is open or not:

```
result%=IS_OPEN(#chan%)
```

where result% is 1 if the channel is open, 0 if not. Please note that the '#' is necessary! If you leave it out, or leave out the channel altogether, the function automatically checks channel 0.

The reasoning here is as follows: if you are running the program in the SBasic interpreter, then (in all normal cases), window#0 is the first window opened for this job (i.e. this copy of the SBasic interpreter). So we can check whether this window is open. If it is, then the outline of window is set. You will notice that this code will work in programs running under the interpreter (whether they have been EXEC'ed, or loaded and run) and programs compiled under QLiberator.

If window#0 is not open, than all is well, it is presumed that NO window is as yet open for this program. In this case, the PWactivate keyword automatically sets the outline for the window it will open when activating the outline object. This test will fail if, for any reason, you closed window#0, but I can't fathom why you would want to do that under the interpreter!

If window#0 is open, then we must specifically set the outline for this window before we activate our Prowess outline object. However, as I mentioned above, all windows of a program MUST fit within the outline of the first window opened for this program (which is often called the "primary window"). Thus, it doesn't make sense to set the outline of window#0 to a small value, since our Prowess window could not be bigger than that. Ideally then, we would make outline of window#0 as big as possible (i.e. so that it covers the whole screen), except that I want to leave a space of 28 pixels on the top of the screen, so that I can continue to see the buttons I place on top of the screen. 28 pixels leave enough space for two rows of normal buttons.

So, we want to make the outline of window#0 as large as the screen is, and as high as the screen is minus 28 pixels. What are the screen dimensions, then? This is not obvious anymore. It used to be that a QL screen (in mode 4) was 512 pixels wide by 256 pixels high, but with the advent of the QXL, Aurora, QPC and other Atari emulators, the size of the screen can vary enormously - I'm writing this on a system where the screen is 800 by 600 pixels.

So we first must find out what the screen area is like. You can, of course, presume that the screen will always be exactly the size of YOUR screen, but if your program is to run on all kinds of sys-

tems, then this is not true. We need some (universal) way of finding out how large the screen can be. This can be found out with the PWscrsize keyword, as used in line 390 of the example program. The PWscrsize command will tell you what the maximum screen size for a window can be:

```
PWscrsize [#chan%,]x_size%,y_size%,x_orig%,y_orig%
```

The four parameters (i.e. apart from the optional #chan%) will be filled in **ON RETURN FROM THIS KEYWORD**, so they can have any value when you call this keyword - but they must be of the correct type, i.e. integer variables.

- -› ***#chan%*** is the channel ID. Normally this would be channel #0. You can leave this parameter out, it will then default to channel #0.
- -› ***x_size%*** and ***y_size%*** will be set to the maximum x and y sizes your window can extend to. Normally, if this is done for window#0, i.e. the first window for a job, this should be as large as the screen itself. For all other windows opened for the same job, this should be the x and y sizes of the outline of window#0.
- -› ***x_orig%*** and ***y_orig%*** will be set to the x and y origins of the screen (i.e. 0 and 0 for the first window open for a job).

Please note that the parameters passed MUST be integers, and must be variables.
So, in line 390, we get the maximum size window#0 can occupy, and since it is presumed that window#0 is the first window open for this job, this is actually the size of the screen.
In line 410, we then set the outline for window#0 to cover the entire screen, minus a row of 28 pixels at the top. Then we set windows#0,#1,#2 to their normal aspects in SBasic. Please note that the outline of window#0 is NOT affected by the size of window#0 itself. The outline of a window, and the size of that window are two different things: the size is what you see of the window on the screen, the outline is the size the Pointer Interface "knows" and "cares" about.

The 'init' procedure

So now we have made sure that, if necessary, the outline of the first window open for this job has been set. This was done by calling the set_windows procedure from within the init procedure (in line 170). Now we go on to create the outline object of the window, with the following code in line 210:

```
210 outline=PWcreate(0,PW('TYPE_OUTLINE'),
PW('OUTLINE_TITLE_TEXT'),text$,PW('OUTLINE_QUIT'),
PW('OUTLINE_QUIT_KEYPRESS'),27)
```

(please note that this is all on one line).

Let's examine that: First of all, we pass a parameter of 0. This is the owner of the object, and, since this is the very first object we create, there is no owner for it yet. Then we pass a tag stating the this is going to be an outline object (PW('TYPE_OUTLINE')).

The next tag is a tag specifically for outline objects, and its purpose is to tell the outline object that it will have a title with a text in it(PW('OUTLINE_TITLE_TEXT')), and the text then follows (e.g. the variable text$, initialised in line 190 - please note the way there is a CHR$(0) tucked in at the end. This is necessary, and will be explained later).

Next, we tell the outline that it is to have a Quit item (PW('OUTLINE_QUIT')) and that this quit item is to be indicated (hit/done) with a certain keypress - in this case the ESC key, which corresponds to CHR$(27).

That's it for initialising the outline object. Now we have to activate it. This is done in the 'main' procedure.

The 'main' procedure

Here, we first of all initialise some variables (line 260), e.g. most of those that will be passed on to the PWactivate procedure. This initialisation is not strictly necessary (most of the will be set upon return from this function) but is good programming practice, and for the "mempointer" variable this is absolutely necessary.

Then we activate the outline object (line 280). Now the code passes to Prowess, it will draw the pointer and read the events.
Since the routine is more or less empty (apart from the title and the quit item), nothing much happens. You can quit the window by pressing the ESC key.

We then arrive at line 290. Mempointer will be 0 since we wanted to quit this program, so we leave the loop and remove the outline object (which otherwisewould still exist and eat up memory).
Admittedly, this program doesn't do very much - yet. We can later build up on it but before that, we'll see two new concepts - the 'PW' function, and string handling under Prowess. This will be covered in the next instalment of this series.

QL Forever!

BASIC Linker - a Review

Tim Swenson

BASIC Linker is a programmers tool for writing SuperBasic programs. It is written by Wolfang Lenerz and distributed by Jochen Merz Software.

Some Background

One of the key differences between BASIC and other languages (like C, Pascal, Perl, etc.) is that BASIC was designed to have line numbers, GOTO, and GOSUB statements. A lot of early BASIC's were written for "home computers" where the BASIC interpreter was the operating system for the computer. Examples of these systems are Atari, Commodore, Apple, and Sinclair. The QL was the first Sinclair computer that had a "real" OS, but it still included a built-in BASIC interpreter.

These BASIC interpreters put a lot of constraints on the language and for some, these constraints were considered part of the langauge. One of the constraints was how BASIC programs were edited. Most computers had a form of line editor built-in as part of the interpreter. Some options were added to these built-in editors, like the EDIT and ED on the QL, but they were still a constraint.

To those used to programming in different languages and on different systems, the practice of programming was done with a standard editor (the programmers' choice) and a compiler/interpreter. BASIC Linker is a tool that brings that form of programming to the QL.

What is BASIC Linker?

In the Unix sense, BASIC Linker is a filter. In the C sense, it is a preprocessor. In the SuperBasic sense, it is a program that takes a SuperBasic program that does not include line numbers, and adds these numbers and creates an output file. This is the base function of BASIC Linker, but it does more than just this.

It allows a single program to be made up of many files. When BASIC Linker is run, it reads all these files, adds the line numbers and creates one big output file with the single program in it.

BASIC Linker can also parse a program (without S*BASIC) and find syntax errors. It can also execute Qliberator and compile a program for you. When used to its fullest potential, you can edit a program in QD, save it, run it through BASIC Linker, which checks the syntax, and then executes Qliberator to produce an executable. This saves you the time of having to run Qliberator yourself.

Requirements

BASIC Linker requires the Pointer Environment and the Menu Extension, both of which come with it. It is designed to operate with QD as the editor. It can be run without an editor, but trying to use some of the features that are designed for QD will cause BASIC Linker to die with a QLIB error.

Programming for BASIC Linker

Since there are no line numbers when programming with BASIC Linker, two line number oriented SuperBasic commands are not allowed (GOTO and GOSUB) and RESTORE is altered. A further limitation is put on how Function and Procedure definitions are written.

Losing GOTOs and GOSUBs is not a big problem. With the advent of Procedure based BASIC's (like SuperBasic), GOTOs and GOSUBs are no longer needed. For some programmers, GOTOs are a bad thing (see Dijkstra's famous paper entitled "GOTO Statement Considered Harmful" - **www.acm.org/classics/oct95/**).

RESTORE is used in conjuction with a DATA statement, telling SuperBasic to start reading data from certain DATA statements, starting a specific line number. With BASIC Linker, the line number is never known to the programmers, the workaround is to place a RESTORE statement before each group of DATA statements and the RESTORE will be given a line number of itself.

Here is an example taken from the manual:

```
100 RESTORE 110
110 DATA ..........
```

In BASIC Linker

```
RESTORE
DATA.....
```

Which will become the following lines,

```
100 RESTORE 100
110 DATA ......
```

If you are a big user of DATA statements, some experimentation may be necessary to understand how BASIC Linker can produce the code you need. The end code (with line numbers) can be checked to see what it looks like.

There is a limitation on how Procedures and Functions are defined. Basically (no pun intended), colons are not allowed in a statement defining a Procedure or Function.

The following is not allowed:

```
  DEFine PROCedure abc:
next command : END DEFine
abc
  DEFine FuNction abc:
next command :  END
DEFine abc
```

The DEFine and END DEFine must be on separate lines.
Personally I have no problem with this limitation as I do not like to use colons to combine commands in a line. The only time I may do it is in short commands that are related (INK, PAPER, CLS, etc).

The Linker Control File

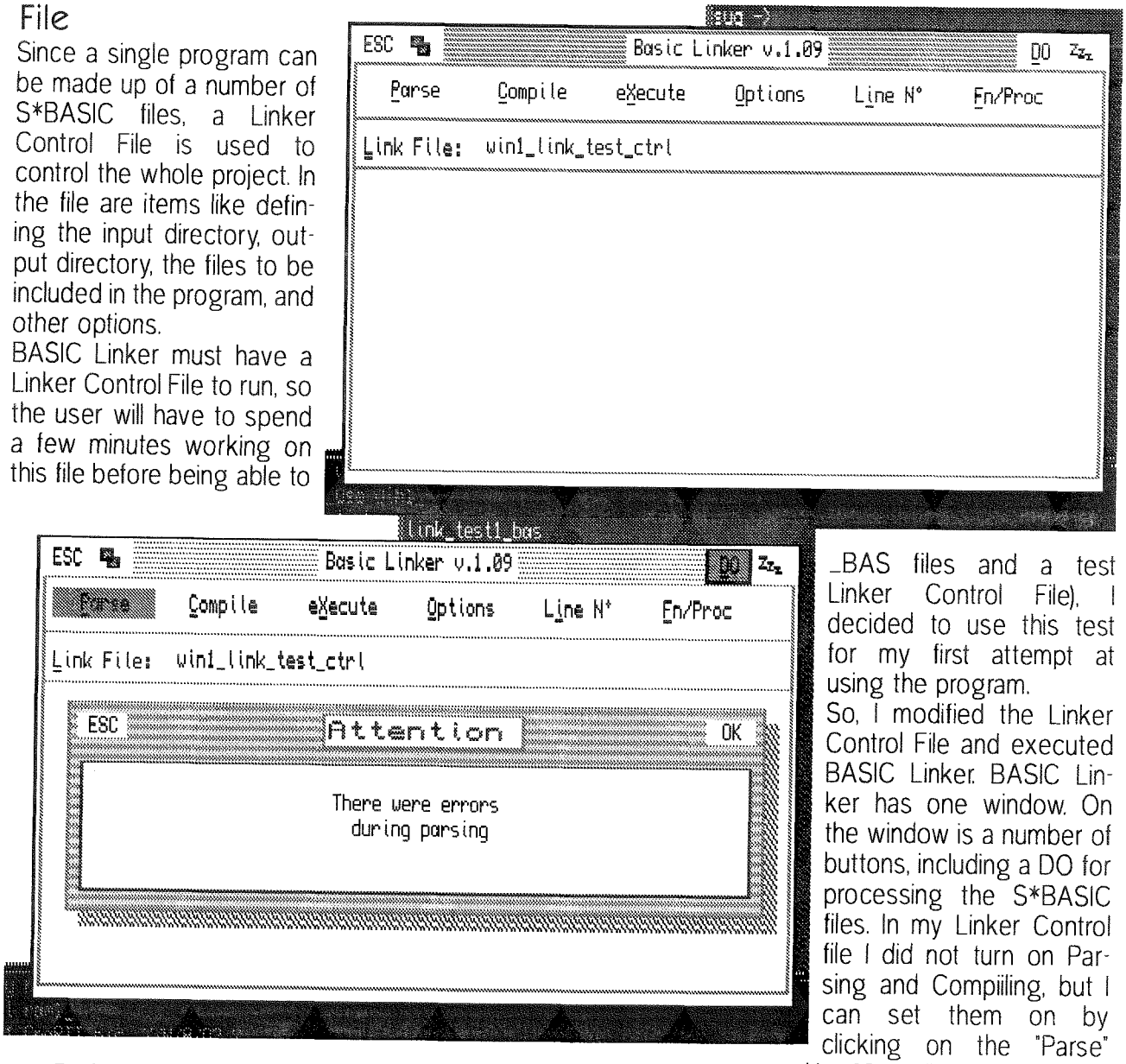
Since a single program can be made up of a number of S*BASIC files, a Linker Control File is used to control the whole project. In the file are items like defining the input directory, output directory, the files to be included in the program, and other options.
BASIC Linker must have a Linker Control File to run, so the user will have to spend a few minutes working on this file before being able to run BASIC Linker. A sample Linker Control File is provided, so it is easiest to copy this and start from there.
Users new to the program will spend most of their time learning how to change the options and control of the program via the Control File. Creating a complex program Control file can also take some time due to the the large number of options and control statements available.

Configuring BASIC Linker

BASIC Linker has a Level 2 Config Block. It can only be configured with MenuConfig and not the standard Config program (Config only does Level 1 Config Blocks). MenuConfig does come on disk with BASIC Linker. It took me a minute to realize that I had to use MenuConfig, as I have always used Config in the past.
Before running BASIC Linker, I had to configure the directory where it is to find the Linker Control Files, where it will find QD, and where it will find QLiberator. There are other options but I did not need to change them.

Running BASIC Linker

Since BASIC Linker comes with a program example (three _BAS files and a test Linker Control File), I decided to use this test for my first attempt at using the program.
So, I modified the Linker Control File and executed BASIC Linker. BASIC Linker has one window. On the window is a number of buttons, including a DO for processing the S*BASIC files. In my Linker Control file I did not turn on Parsing and Compiiling, but I can set them on by clicking on the "Parse" and/or "Compile" buttons. The first thing I did was to just hit "DO". A couple of smaller popup windows appeared and quickly disappeared before I could read them, then I was back to the original window. I guess no news is good news.
I CTRL-C'ed back to SBASIC and looked at the resultant file.



It was there and it had line numbers in it. So, it looks like the program worked. When I tried this again, I noticed that the small popup windows are progress windows, telling the user what BASIC Linker is doing. Since I'm using a Q40 and the test programs were real small, the popups did not get a chance to stay on the screen before the next progress popup was to appear.

Now to try Parse and Compile. CTRL-C'ed back to BASIC Linker, clicked on "Parse" and "Compile", then hit DO. Some stuff happens and then the QLiberator window quickly pops up and then goes away. Back to SBASIC I can see that there is a "test_obj" that I then execute and see that it does run.

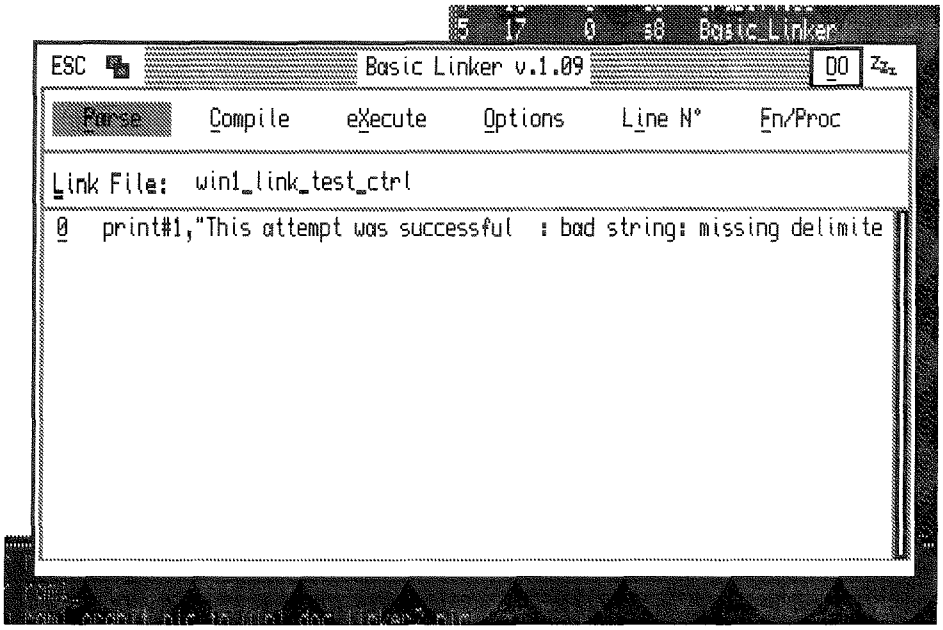
So, instead of doing all of the QLiberator stuff myself, BASIC Linker has done it and very quickly generated an executable. I did forget to mention that one additional button that I did not use is "Execute". With this turned on, once the program in compiled, BASIC Linker will execute it so that it can be tested.

What Happens With Bad Code

Know that I knew what BASIC Linker does when everything is fine, time to test it with bad code. I edited one of the test files and removed the last quote (") from the end of a print statement and saved the file.

Back to BASIC Linker, clicked off Parse and Compile, then hit DO. Same as before. When I CTRL-C'ed back to SBASIC and VIEWed the file, I did see the quote character was still missing. Now back to BASIC Linker, click on Parse, and then DO. This time I see the small popup window stay on the screen saying "There was an Error in Parsing". I click on OK, the window disappears, and I'm back to the main BASIC Linker window. But, I see that the line with the missing quote is listed along with the error message. I click on the error message (DO it) and suddenly I'm back to QD and I can see BASIC Linker executing a QD command to put the cursor on the line with the bad code.

This is probably the best feature of BASIC Linker. Instead of CTRL-Cing back to an editor and finding the offending line, BASIC Linker does it for you. As simple as it may sound, it does save a lot of keystrokes and eventually a lot of time.



Once in QD, I edited the line, saved the file, and CTRL-Ced back to BASIC Linker to try it again.

More Options

Since BASIC Linker executes QLiberator, it has a part of its Config Block where options for QLiberator are defined. Things like No Windows, No Lines, etc. This is configured for all of BASIC Linker and not per program that is going to be processed and compiled.

If there were any compiler errors, it would take you back to them in the editor. Once edited, the compile command could be executed to try it again.

BASIC Linker gets us almost there. It does have the ability to get to QD and get to a line with bad code, but I would still like to run BASIC Linker from within QD. Now, I'm not a QD expert and had to borrow a copy for this review, so I don't know if this is possible or not. I did not see any mention of it in the BASIC Linker manual.

Summary

Since I did not have any programming projects I was working on, I did not go much further than my initial tests. I did not try to break the program. The program is currently being supported and any bugs that arise should quickly be fixed. In fact, while doing this review, an updated version of BASIC Linker was sent to me.

The one thing that the QL has really been lacking is a good Program Development System. One of the first and best that I've used came with Turbo Pascal (on both DOS and CP/M). It was a package of integrated editor and compiler.

QUANTA

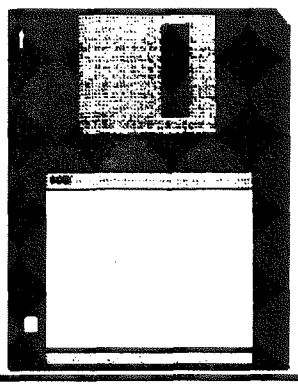
Independent QL Users Group

Worldwide Membership is by subscription
only, and offers the following benefits:
Monthly Newsletter - up to 40 pages
Massive Software Library - All Free !
Free Helpline and Workshops
Regional Sub-Groups. One near you?
Advice on Software and Hardware problems
Subscription just £14 for UK members
Overseas subscription £17

Barclaycard: Visa: Access: Mastercard

*** Now in our SEVENTEENTH successful year ***

Further details from the Membership Secretary



Bill Newell
213 Manor Road
Benfleet
Essex
SS7 4JD
Tel. (01268) 754407

It would also be nice if BASIC Linker could be configured to work with more that just QD. QD is nice, but there are also some nice Freeware editors out there. My personal favorite is MicroEmacs, and now that MicroEmacs is a CSM server, having BASIC Linker send commands to MicroEmacs to move to a specific line would be fairly easy to do.
But still, BASIC Linker does free up the S*BASIC programmer from the confines of line numbers and the limited built-in editors in SuperBasic and SBASIC. There is a significant learning curve when creating Linker Control files, but once one is created, BASIC Linker can save many keystrokes by quickly being able to re-compile a program after fixes are made.
When programming, I find that I'm spending a lot of my time fixing syntax errors and programming errors. This includes running QLiberator numerous times in one programming session. I can automate things by using ALT keys, but BASIC Linker makes it easier.
Programming this way may not be for everyone. Some programmers may like using the built-in editing capabilities. For those that don't and like to choose their editors, BASIC Linker makes programming easier.

Programming in Assembler - Part 8

Norman Dunbar

We carry on exactly where we stopped last time.

Getting Parameters

On entry to a machine code extension (ie not an EXEC'd job or a CALLed routine) certain registers are set up with very useful values. These are:

Register	Value
A1	ALLEGEDLY points to the top of the maths stack relative to A6.
A3	Points to the start of the name table entry for the first parameter.
A5	Points to the first byte AFTER the name table entry for the last parameter.
A6	Base address of SuperBasic DO NOT CHANGE THIS REGISTER.

A1 is supposed to point at the top of the maths stack (see below) relative to A6, but I have found out the hard way that this is only the case when the procedure or function being executed has some parameters and they have been fetched. A1 is set to the amount of space used (or free) on the maths stack on entry to a procedure. (See Maths Stack below for full details.)

A3 points at the address of the first byte of the first entry in the name table for this procedure or function. Again, this is relative to A6.

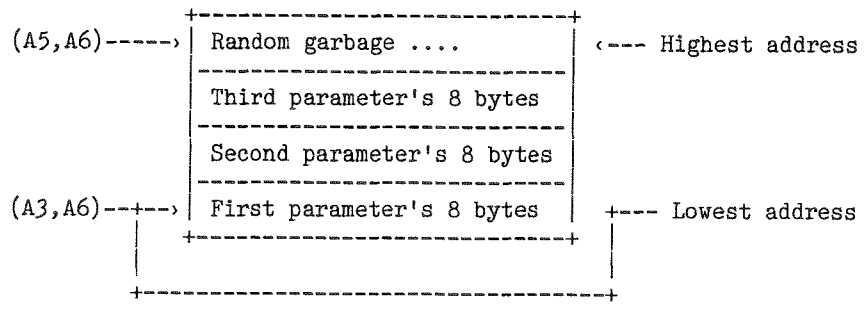
A5 points at the address of the first byte AFTER the last name table entry for this procedure or function. Again this is relative to A6.

A6 should never be changed as it points to the base of the SuperBasic job and almost all the various routines involving the maths stack and getting/returning parameters rely on addresses being relative to A6.

So we can now check to see how many parameters we have by the following calculation:

```
(A5 - A3) / 8
```

There are 8 bytes in each name table entry. Full details of the name table entries are given below.
If we have 3 parameters, then the name table entries will look like the following:



So (A3,A6) points to the first byte of the first parameter and is the lowest address, (A5,A6) points to the first byte past the last parameter and is the highest address.

The first name table entry starts at 0(A3,A6) and ends at 7(A3,A6). The second starts at 8(A3,A6) and ends at 15(A3,A6) and the last starts at 16(A3,A6) and stops at 23(A3,A6).

When fetching parameters from the name list onto the maths stack, we can use some vectored utilities to get them for us. These allow the retrieval of strings, long words, integers (short words) and floating point values. They all expect A3 and A5 to be set up correctly as above. A3 and A5 are trashed by the routines, so if you have to check any parameter separators etc, then you must do it before calling the fetch routines.

When the routines return, they set D3.W to the number of parameters fetched and set A1 to the correct value for the top of the maths stack - relative to A6 of course. Now we can access the values of each parameter separately as we like. On return the first parameter in the list is stored at 0(A1,A6), the next is above the first and so on. When fetching parameters p1, p2, p3 from a procedure or function call, they will end up on the maths stack in the correct order - 0(A1,A6) will be pointing at p1 on the stack.

The parameter fetching routines are:

CA_GTINT - fetch integer parameters (2 bytes each)
CA_GTLIN - fetch long parameters (4 bytes each)
CA_GTFP - fetch floating point parameters (6 bytes each)
CA_GTSTR - fetch string parameters (variable length)

They require to be called as follows:

```
        move.w  ca_gtint,a2     ; Fetch all params as integer words
        jsr     (a2)            ; Do it
        tst.l   d0              ; Did it work ?
        beq.s   ok              ; Yes
        rts                     ; Must return to SuperBasic with errors

ok                              ; carry on here
```

At this point, D3.W can be tested to check that the correct number of parameters has been fetched.

```
        cmpi.w  #4,d3           ; Were there 4 parameters ?
        beq.s   ok_4            ; Yes
        moveq   #-15,d0         ; Bad parameter error code = -15
        rts                     ; and back to SuperBasic

ok_4                            ; Carry on here
```

To access the parameters we need to get the data off of the maths stack and into our working registers, as follows:

```
        move.w  0(a6,a1.l),d1   ; Parameter one
        move.w  2(a6,a1.l),d2   ; Parameter two
        move.w  4(a6,a1.l),d3   ; Parameter three
        move.w  6(a6,a1.l),d4   ; Parameter four
```

and so on. Now that we have our parameters, we need do nothing more with the maths stack if we are inside the code of a procedure. If we are in a function then we MUST tidy the maths stack. This is simply done by adding the size of all parameters on the stack to A1. In our example we have 4 word length parameters, so we should add 8 to A1 as follows:

```
        adda.l  #8,a1           ; Reset maths stack
```

As mentioned, there is no need to do this in a procedure, but if you have to learn to do it for a function, you are as well to learn to do it for everything - that way you don't forget to do it and cause a hanging QL.

Tidying a stack with strings on is more difficult and it is probaly best done as each one is removed. For example, say we have two strings on the stack after a call to CA_GTSTR then we get them off as follows:

```
        cmpi.w  #2,d3           ; Were there two strings?
        beq.s   ok              ; Yes
        moveq   #-15,d0         ; Bad parameter
        rts                     ; Exit to SuperBasic

ok      lea     buffer_a,a2     ; Destination for one string
        lea     0(a6,a1.l),a3   ; Source for string
        bsr     copy_str        ; Copy
        move.w  0(a6,a1.l),d0   ; Size word
        addq.w  #3,d0           ; Make bigger
        bclr    #0,d0           ; Make even
        add.w   d0,a1           ; Add D0 to A1 - sign extends remember !
```

Ok, so we added the size of the first string plus 2 for the size of the size word as well, to A1 having made it even so the stack is now cleared of the first string. This leaves one string with its size word sitting at 0(A6,a1.l) ready for the next copy:

```
        lea     buffer_b,a2     ; Destination for next string
        lea     0(a6,a1.l),a3   ; Source for string
        bsr     copy_str        ; Do the copy
        move.w  0(a6,a1.l),d0   ; Size word
        addq.w  #3,d0           ; Make bigger
        bclr    #0,d0           ; Make even
        add.w   d0,a1           ; Add D0 to A1 - sign extends remember !
```

and there you have a tidy stack once again.

You could ask 'if we have to restore A1 to its value on entry, why not just save A1 and then restore it afterward?'. Like this:

```
start   move.l  bv_rip(a6),a1   ; Fetch top of Maths Stack
        move.l  a1,-(a7)        ; Stack it for later

        ; Do lots of stuff here - fetching parameters etc

        move.l  (a7)+,a1        ; Restore A1

        ; and so on
```

Well, you could, but at certain times there will be a hung QL and you will not know why. The reason is simple, but difficult to find or trace. When you fetch parameters onto the maths stack, it can MOVE IN MEMORY. Preserving the original value is fine if the stack stays put, but if it moves and you set BV_RIP to the old value, you can get into all sorts of trouble. It is best to keep the stack tidy using the methods described above.

Keeping Things Even

You may well also ask "What is all this add 3 and clear bit 0 nonsense then?" Think about it in binary for a bit. We have the word size of the string in D0.W and we must ensure that we add an even number of bytes to A1. We must also remember to add 2 to A1 for the size of the size word itself.

RWAP SOFTWARE

QL Cash Trader v3.7 £5

A well established accounts package for the small to medium sized business, including automatic generation of profit & loss account, balance sheet, VAT return, reports and analysis for audit trails and management decisions. Previously sold for over £100.*

QL Payroll v3.5 £5

Manage a payroll for a small to medium sized business. Handles up to 99 employees easily, producing P45s and P60s as well as the payslips on a monthly or weekly basis. Calculates tax and national insurance and is easy to update to take account of current tax year rules.

Q-Help v1.05 £10
Q-Index v1.04 £5

Q-Help: on-screen help for SuperBASIC cmmands, including TK2, Turbo Toolkit, SMSQ/E and PD toolkits. Can easily be used to add help pages to your own programs - simply produce ASCII text for each help page, an index to the help pages and Q-Help automatically cross-references and displays the links. The PD toolkits referred to are available for £2.
Q-Index: The SuperBASIC index supplied with the Reference Manual - enter a topic such as 'screen resolution' and find out the commands which relate.

Sidewriter v1.08 £10

Produces landscape printouts of Easel/QSpread spreadsheets and output from QL Genealogist, as well as any other standard text file. You can specify the fonts to be used on the page. Works with all EPSON compatible printers, from 9 pin dot matrix up to inkjet printers. A most useful utility written by Dilwyn Jones - you know it must be easy to use.

ProForma ESC/P2 Drivers v1.01 £8

New improved printer drivers, providing up to 720 dpi printout in full colour from all programs written for use with ProWesS, such as LineDesign and Paragraph. Work on all Epson inkjet printers which support binary mode compression (740,850 and 900 models at least). 1440 dpi to follow.

QL Genealogist v3.25 £20
Genealogy For Windows £50

Keep track of your family tree! Add individuals, with details of their parents and children, watch all of those links build up into a formal family tree layout. Text files and Pictures may also be linked to individuals as well as notes and events, making this the perfect way to preserve the history of your family for future generations. QL version nnw supports FileInfo II and QMenu, as well as keeping details of both the male and female trees. PC version is event driven - enter the details as they appear in documents and it generates the tree from these. Both programs easy to use with step by step tutorial.

D-Day MKII v3.04 £10
Grey Wolf v1.8 £8
War In the East MKII v1.24 (Upgrade Only) £10

For the wargaming enthusiast - D-Day is a classic table top wargame, where you control either the Allies or the Axis forces and play against either the computer or another human player. With the ability to define your own army set ups and a choice of four different scenarios, this should keep you entertained for a while. Grey Wolf places you in charge of a submarine - can you sink the enemy shipping whilst avoiding their planes and destroyers??

FlashBack SE v2.03 (Upgrade) £5

This is the ultimate database program - extremely fast and flexible, easy to use, updated to cope with the latest versions of the QL operating system and still maintained. A report module is included to allow you to format output in any way, including mail-merge. Unfortunately, only available as an upgrade to the original version.

SBASIC/SuperBASIC Reference Manual £40
Updates £6 each, £10 for 2 (Current Version - Rel 3)

Have you ever tried to write a program, but been lost as to the means of performing a certain action? This Reference Manual provides you with a full description and examples of how to use all of the keywords found on a standard QL, plus the keywords under SMSQ/E, Toolkit II and many different public domain toolkits. Details of any possible problems are provided, together with descriptions of how to use the device drivers and how to ensure that your programs are compatible across the range of QL platforms.

This book is ideal for all QL users and is kept up to date by regular updates.

Orders are currently being taken for the next print run of this popular tome.

(Note Price for the book does not include postage and packing).

Return To Eden v3.08 £10
Nemesis MKII v2.03 £8
The Prawn v2.01 £8
Horrorday v3.1 £8
West v2.00 £5
The Lost Kingdom of Zkul v2.01 £5

Classic QL adventures, now re-released without any need for microdrives. These include mainly text adventures, catering for all tastes, from the spoof Prawn, through to a Hammer Horror, fighting the bad-guys in the old West and battling with robotic hoards and goblins. Return to Eden is a massive three disks of adventure, with pictures for each location and a captured prince to rescue. With three characters to control, each with their own abilities and skills, this one should keep you amused for many an evening.
All six adventures are available together for only £25.

FlightDeck v1.04 £10

Can you learn to fly a twin-engined passenger jet? This simulator includes full shaded 3d views of the world around which you are flying, toegether with the ability to add navigation beacons, airports and even landscape features to make FlightDeck the ultimate QL Flight simulator. A database of the main UK airports is included to allow you to fly around the UK for training missions.

Q-Route v1.08C £25

This is the latest version of this popular route finding program. Find the quickest route or the shortest route between any two places, using roads. A wide ramge of maps is available for this program. (see elsewhere in this advert). The program is easy and quick to use. You can even add your own places and roads to the maps to include local detail.

QL Cosmos v2.02 £5

Ever wondered what the stars in the sky looked like 100 years ago? Or maybe you want to learn the constellations and names of what you can see in the sky. This is the program for you - generates pictures of the stars for any given place or time and provides details on those objects. Includes Halley's Comet, the Moon and the Solar System planets.

Open Golf v5.20 £8
QuizMaster II v2.07 £5
Stone Raider II v2.00 £5
Hoverzone v1.2 £5
Deathstrike v1.5 £5

A range of games to keep you amused on the QL. Some are old favourites, like Golf and a quiz program (with over 500 questions). Whilst others are fast, colourful arcade games. Plenty of variation and skill required - what more can you ask for?
All 5 programs £20 only.

Britain.map v1.10 £2
BIG Britain Map (needs 2MB) v2.01 £5
South & West Yorkshire Map v1.04 £2
Ireland Map v1.00 £5
Belgium Map v1.01 £2
Catalonia Map v1.01 £2

These are the latest maps for Q-Route (now at v1.08C). Find your way around the various countries covered. South and West Yorkshire Map is a much more detailed area of that beautiful part of the British isles.

RWAP Software, 4 Anvil Crescent,
Coseley, West Midlands
WV14 8GA

TEL: 01902 836888

Cheques in £sterling payable to 'R.Mellor'

* Also known as Trading Accounts

Lets try this with an even number first of all. Even numbers are detected by bit zero being clear, so:

D0	D0 + 3	Result
2	5	4
4	7	6
10	13	12

So you can see what is happening. D0 always ends up being D0 + 2 and is always even. This is good as it is what we want. What about odd numbers then?

D0	D0 + 3	Result
3	6	6
5	8	8
11	14	14

So is this good then? Remember that the maths stack must be kept even. When odd length strings are copied onto it by CA_GTSTR it pads out the space on the stack with a rubbish byte (CHR$(0) to be precise) which is never used. The size word remains odd.

So for an odd sized string we need to add 2 for the size word, the odd number of bytes and one spare for the padding. Our 3 lines of code handle this for all cases - even or odd sized strings. The code is good!

Of course it would be simple to do this:

```
        move.w  0(a6,a1.l),d0   ; Size word
        btst    #0,d0           ; Is it even ?
        beq.s   even            ; Yes
        addq.w  #1,d0           ; Add 1 for padding byte for odd sized strings

even    addq.w  #2,d0           ; Add 2 ro the size word
        add.w   d0,a1           ; Add D0 to A1 - sign extends remember !
```

But this is extra typing and takes longer, so the simple case shown above, works all the time.

Two of These and One of Those

What do you do if you want to get hold of two long words and a string?

Let us assume that you are writing a procedure that has this format:

```
DO_SOMETHING long_1, long_2, string_1
```

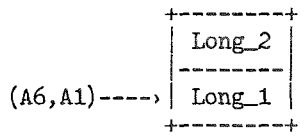
This has two different types of parameters and we cannot fetch them all in one go unless we can read the long parameters as strings and convert them ourselves. It is quite easy to fetch these parameters - you just do it in two goes.

In the code we know that A3 and A5 hold the start and stop addresses of the parameters in the Name Table. If we set A5 to be A3 + 16 and then collect long words we will get our two long words. We can then set A5 back to its original value and set A3 to this less 8 and fetch the final parameter as a string. Here we go then:

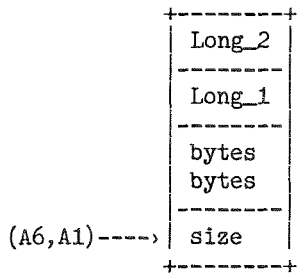
```
get_longs   move.l  a5,-(a7)        ; Save last parameter pointer
            lea     16(a3),a5       ; A5 now thinks thare are only two parameters
            move.w  ca_gtlint,a2    ; Fetch all parameters as longs
            jsr     (a2)            ; Do it
            tst.l   d0              ; OK ?
```

```
            beq.s   got_long        ; Yes
            rts                     ; Exit with error code

got_long    cmpi.w  #2,d3           ; Were there two ?
            bne.s   bad_params      ; No, bale out
            move.l  (a7)+,a5        ; A5 holds address of final string
            lea     -8(a5),a3       ; Pretend we have only one parameter
            move.w  ca_gtstr,a2     ; Fetch as strings now
            jsr     (a2)            ; Do it
            tst.l   d0              ; OK ?
            beq.s   got_string      ; Yes
            rts                     ; Exit with error code

bad_params  moveq   #-15,d0         ; Bad parameter error
            rts                     ; Exit to SuperBasic
got_string  ; continue from here
```

Ok, so now what does the maths stack look like? Remember when fetching parameters they end up on the stack in the order you want them with the first at the lowest address and the next above it and so on. This time, we fetched two longs and a string in two different calls. This means that after the first fetch the maths stack is:



But then we fetched a string and it got put onto the maths stack so it now looks like this:



QDOS is very helpful here. If during the course of fetching the string, the maths stack had to be moved in memory, QDOS will preserve the current contents so that 'long_1' and 'long_2' will still be there when you come around to using their values. Nice!

In this discussion we mentioned the name table. This is discussed in detail next. Do you get the feeling that this article is written upside down?

Name Table Entries

The name table is a list of 8 byte entries which define all the names used in SuperBasic (or extensions to SuperBasic written in assembler), the type of each entry and where it lives in the name list and the SuperBasic variables area.

As per the description above (GETTING PARAMETERS), the name table is also used to store details of the parameters passed to our assembly routine. So for parameters passed, a copy is made and stored at the end of the name table. The A3 and A5 registers are set up to point at the first and last parameter and for these, the format of the name table is as follows:

Bytes 0 & 1	Bytes 2 & 3	Bytes 4 to 7
Type & Separator flag word	Pointer to NAME LIST entry MAY BE AN ODD ADDRESS	Pointer to value in variables area

The low byte of the type word tells us what type of parameter we are dealing with and its separator(s) as follows:

Bit 7	1 = There is a hash (#) in front of this parameter. 0 = There is not a hash.
Bits 6-4	000 = No separator after this parameter 001 = Comma (,) after this parameter 010 = Semi-colon (;) after this parameter 011 = Back-slash (\) after this parameter 100 = Exclamation mark (!) after this parameter 101 = TO after this parameter
Bits 3-0	0000 = null 0001 = string 0010 = Floating point 0011 = Integer

For the first parameter, the type byte is at 1(a6,a3.l) as opposed to 0(a6,a3.l).

For the rest of SuperBasic, the name table uses bytes 0 and 1 to define the type of the entry as follows:

Byte 0
$00 = Undefined $01 = Expression $02 = Variable $03 = Array or substring $04 = SuperBasic PROCedure (Byte 1 is always zero) $05 = SuperBasic FuNction

Byte 1
$00 = Substring (Internal use only !) $01 = String $02 = Floating point $03 = Integer

Both Together
$0602 = REPeat loop identifier $0702 = FOR loop identifier $0800 = Assembly language procedure $0900 = Assembly languge function

Note:
The REPeat and FOR loop identifiers are hard coded to be of type floating point. This represents tha internal values for SuperBasic. I suspect that this is the reason that FOR loop identifiers cannot be integer. I believe that SBASIC allows integer FOR loops and I presume that the internal format for these will be $0703 - I am sure that Jochen will correct me if I am wrong!!

All entries in the name table, be they parameters or 'proper' names, have a word in bytes 2 & 3 which points to the entry in the name list for this 'name'. This simply gives an easy way of storing the names all in one place. Note that this value is simply the offset from the start of the name table where the bytes of this name can be found. A fuller description of the name list follows on (in the best tradition of upside down magazine articles!) below.

If the value is -1, then this is an expression and has no name.

TF Services

superHermes

A major hardware upgrade for the QL

All Hermes features (working ser1/2, independent baud rates/de-bounced keyboard/keyclick) PLUS full 19200 throughput on ser1/ser2 not affected by sound IBM AT keyboard interface (plus foreign drivers) // HIGH SPEED RS232 industry standard two-way serial port. 4800cps throughput (supergoldcard - qtpi - zmodem) at 57600bps // THREE low speed RS232 inputs (1200 to 30bps)- incl SERIAL MOUSE driver. Other uses include RTTY/graphics tablet etc // 3 spare I/O lines (logic) with GND/+5V // Capslock/ scrollock LED/Turbo/keylock connectors // 1.5k user data permanently in EEPROM

All this on a professional board about twice the size of the 8049 co-processor it replaces

Cost (including manual/software). **£90** (£92/£87/£90)
IBM AT UK layout Keyboard..... **£22** (£24/£23/£27)
Serial mouse ..... **£11** (£13/£12/£14)
Capslock/scrollock LED ..... **£1** (£1.50/£1/£1.50)
Keyboard or mouse lead ..... **£3** (£3.50/£3/£3.50)
High speed serial (ser3) lead ..... **£4** (£4.50/£4/£4.50)

superHermes LITE

All Hermes features (see above) + an IBM AT keyboard interface only. Entry level superHermes.
Cost (incl keyboard lead)...**£53** (£55.50/£51/£53.50)

Minerva

MINERVA RTC (MKII) + battery for 256 bytes ram. CRASHPROOF clock & I²C bus for interfacing. Can autoboot from battery backed ram. Quick start-up.

The ORIGINAL system operating system upgrade

OTHER FEATURES COMMON TO ALL VERSIONS

DEBUGGED operating system/ autoboot on reset of power failure/ Multiple Basic/ faster scheduler- graphics (within 10% of lightning) - string handling/ WHEN ERROR/ 2nd screen/ TRACE/ non-English keyboard drivers/ "warm" fast reset. V1.97 with split OUTPUT baud rates (+ Hermes) & built in Multibasic.

First upgrade free. Otherwise send £3 (+£5 for manual if requd). Send disk plus SAE or two IRCs

MKI...£40 (£41/£40/£43) **MKII...£65** (£66/£63/£67)

QL REPAIRS (UK only)

Fixed price for unmodified QLs, excl microdrives. QLs tested with Thorn-EMI rig and ROM software.

£27 incl 6 month guarantee

QL RomDisq

Up to 8 mbyte of flash memory for the QL

A small plug in circuit for the QL's ROM port (or Aurora) giving 2, 4 or 8 mbytes of permanent storage - it can be thought of as a portable hard disk on a card, and reads at some 2 mbytes per second. Think of it - you could fully boot an expanded QL, including all drivers/SMSQ etc off **RomDisq** at hard disk speed with only a memory expansion needed.

2 mbytes RomDisq..... **£39** (£41£37/£40)
4mbytes RomDisq..... **£65**(£66/£63/£67)
8 mbytes RomDisq..... **£98** (£100/£95/£99)
Aurora adaptor..... **£3** (£3.50/£3/£4)

MPLANE

A low profile powered backplane with ROM port

A three expansion backplane with ROM port included for RomDisq etc. Aurora can be fitted in notebook case and powered off single 5V rail - contact QBranch for details. Two boards (eg Aurora and Gold Card/Super Gold Card/Goldfire fixed to base. Suitable for Aurora (ROM accessible from outside) & QL motherboard in tower case. Specify ROM facing IN towards boards, or OUT towards back of case.

Cost.....£34 (£36/£33/£35)

I2C INTERFACES

Connects to Minerva MKII and any Philips I²C bus

Power Driver Interface **16 I/O lines with 12 of these used to control 8 current carrying outputs (source and sink capable)**
2 amp (for 8 relays, small motors)..... **£40** (£43/£38/£44)
4 amp total (for motors etc) ..... **£45** (£48/£43/£50)
Relays (8 3a 12v 2-way mains relays (needs 2a power driver)..... **£25** (£28/£23/£27)
Parallel Interface Gives 16 input/output lines. Can be used wherever logic signals are required..**£25** (£28/£23/£27)
Analogue Interface **Gives eight 8 bit analogue to digital inputs (ADC) and two 8 bit digital to analogue outputs (DAC). Used for temp measurements, sound sampling (to 5 KHz), x/y plotting**..... **£30** (£31.50/£29/£30)
Temp probe (-40°C to +125°C)..... **£10** (£10.50/£10/£11)
Connector for four temp probes..... **£10** (£10.50/£10/£11)
Data sheets..... **£2** (£2.50/£2/£3)

QL SPARES

Keyboard membrane..... **£24** (£25/£24/£27)
1377 PAL..... **£3** (£3.50/£3/£4)
Circuit diagrams..... **£3** (£3.50/£3/£4)
68008 cpu or 8049 IPC..... **£8** (£8.50/£7.50/£9)
8301/8302 or JM ROM or serial lead..**£10** (£11.50/£10/£11)
Power supply (sea mail overseas)..... **£12** (£17/£16/£21)

Prices include postage and packing (Airmail where applicable). Prices are: UK (EC,/Europe outside EC/Rest of world). Payment by cheque drawn on bank with UK address, debit card/Mastercard/Access/Eurocard/postal order or CASH! (No Eurocheques). SAE or IRC for full list and details

26 Sep 99

29 Longfield Road, TRING, Herts, HP23 4DG
Tel: 01442-828254 Fax/BBS: 01442-828255
tony@firshman.demon.co.uk http://www.firshman.demon.co.uk

VISA MasterCard

Finally, there is a long word which is the pointer to the variables area. If this value is negative then the variable is undefined and has no entry there. Again, this value is an offset into the variables area and not an absolute address.

Name List

The name list is a simple structure in SuperBasic. It holds the names of all procedures, variables, functions etc that have ever been used in this session at the QL. It is odd in that each name is preceeded by a BYTE defining its length as opposed to a word in the normal QDOS manner. This implies that names can be up to 255 characters long. There are no padding bytes to force even addresses in the name list either. Beware when accessing this area that you only do byte sized operations!

The name list starts at the address BV_NLBAS(A6) to BV_NLP(A6) with BV_NLBAS(A6) being the lowest address and BV_NLP(A6) pointing to the first byte AFTER the last entry in the name list. As usual, the offsets you get from these basic variables are thenselves relative to A6!

To explain further, Fetch the offsets from BV_NLBAS(A6) into A0. The address 0(A6,A0.L) is the start of the name list. Or, in code:

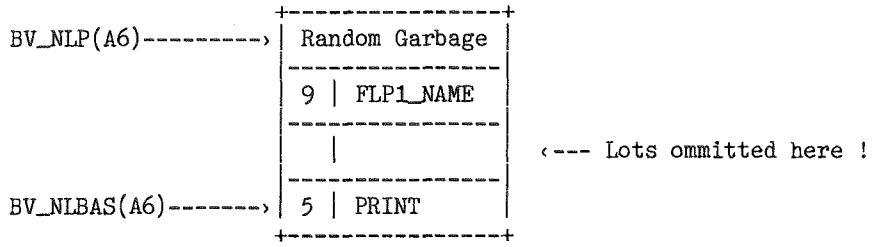
```
start   move.l  BV_NLBAS(a6),a0
        lea.l   0(a6,a0.l),a0
        move.b  0(a0),d0        ; D0 is now the size of the first entry
        ; More code here
```

Now A0 has the start of the name list, but beware of doing this in case SuperBasic gets moved. It is best to stay relative as in the following:

```
start   move.l  BV_NLBAS(a6),a0
        move.b  0(a6,a0.l),d0       ; D0 is now the size of the first entry
        ; More code here
```

This is much safer.

The internal structure therefore looks like this:



How is the name list useful to us in writing procedures and functions? consider these commands:

```
OPEN_IN #3,'ram1_test_file'
OPEN_IN #3,ram1_test_file
```

What is the difference? In the first case, the parameter for the filename is a quoted string and internally, the OPEN_IN routine can fetch it using CA_GTSTR as described above. In the second, it will fail if it uses CA_GTSTR because without quotes, the parameter is a NAME and not a STRING.

The procedure/function writer must check for a string parameter or a name parameter and treat each accordingly. How is this done? - use the name table type byte as described above.

In the procedure or function, process a name as follows:

Assuming that A3 points to the name table entry for this parameter, then if bits 0 to 4 of 1(a6,a3.l) are zero then we have a name and not a variable. We must copy the name to the stack (or to the appropriate buffer) making sure that the size byte in the name list is converted to a size word on the

stack or in the buffer. The following fragment of code gives the general idea:

```
name_test   move.b  1(a6,a3.l),d0
            andi.b  #$0f,d0
            bne.s   not_name
            ;
            ; Must be a name so process accordingly here
            ;
not_name    ; Process a string here
```

So whan a name is detected we have to make space for it, copy the size BYTE from from the name list into the size WORD in our string buffer (which has to be word aligned on an even address) and then copy the individual bytes from the name list to the string buffer. At this point we are in the same situation we would be in had we fetched a string using CA_GTSTR and copied it from the maths stack into our buffer. Simple ? (In my famous DJToolkit extensions I never actually bothered doing this and I simply fetched all filenames etc as strings - if the user supplied a name instead, the procedure or function complained. So far no-one has requested that it be updated to allow names!)

How about a bit of fun - lets write a procedure that prints the entire name list to a channel. It shall be called nlist and it shall take one parameter which is the channel number - this will default to #1 if no parameter supplied.

```
bv_nlbas    equ     $20             ; Base of name list
bv_nlp      equ     $24             ; End of name list
bv_chbas    equ     $30             ; Base of channel table
bv_chp      equ     $34             ; End of channel table
err_no      equ     -6              ; Channel not open error
err_bp      equ     -15             ; Bad parameter error

start       lea     define,a1       ; Pointer to the definition table
            move.w  BP_INIT,a2      ; The vector we need to use (= $110)
            jsr     (a2)            ; Call the vectored routine
            rts                     ; And return any errors back to SuperBasic

*-------------------------------------------------------------------------------
* Definition table for one new procedure
*-------------------------------------------------------------------------------
define      dc.w    1               ; 1 new procedure
            dc.w    nlist-*         ; Offset to procedure
            dc.b    5,'NLIST'       ; Size and name
            dc.w    0               ; End of procedures

            dc.w    0               ; Number of functions
            dc.w    0               ; End of functions

*-------------------------------------------------------------------------------
* Procedure NLIST starts here ...
*
* Check for one or zero parameters - if not then error exit
*-------------------------------------------------------------------------------
nlist       cmpa.l  a3,a5           ; No parameters ?
            beq.s   nl_none         ; Yes, skip
            move.l  a5,d0           ; Last parameter pointer
            sub.l   a3,d0           ; minus first
            cmpi.w  #8,d0           ; One parameter ?
            beq.s   got_one         ; Yes

bad_par     moveq   #-15,d0
error_exit  rts

*-------------------------------------------------------------------------------
* If one parameter, must have a hash else error exit
*-------------------------------------------------------------------------------
got_one     btst    #7,1(a6,a3.l)   ; check for a hash
            beq.s   bad_par         ; Not got one

*-------------------------------------------------------------------------------
```

```
* It has a hash - fetch the channel id. If this fails, error exit.
*-------------------------------------------------------------------------------
get_one     move.w  ca_gtint,a2     ; Vector for word integers
            jsr     (a2)            ; Fetch !
            tst.l   d0              ; Ok ?
            bne.s   error_exit      ; No, bale out
            cmpi.w  #1,d3           ; One only ?
            bne.s   error_exit      ; No, bale out
            move.w  0(a6,a1.l),d0   ; Fetch channel number
            addq.l  #2,a1           ; Tidy stack
            tst.w   d0              ; Set flags
            blt.s   bad_par         ; Negative is a bad channel id
            bra.s   chan_ok         ; skip default channel handling

*-------------------------------------------------------------------------------
* No parameters supplied - default channel number to #1
*-------------------------------------------------------------------------------
nl_none     moveq   #1,d0           ; Default to channel #1
chan_ok     bsr.s   channel_id      ; convert to channel id in A0
            bne.s   error_exit      ; Oops !

*-------------------------------------------------------------------------------
* Fetch the start of the name list from BV_NLBAS(A6). The result of this is
* an offset from A6 to where the namelist actually starts.
*-------------------------------------------------------------------------------
            move.l  bv_nlbas(a6),a3 ; Start of name list (relative a6 !)

*-------------------------------------------------------------------------------
* Our main loop starts here. We test to see if we are finished and if not
* copy the (next) name to the buffer formatting it as a QDOS string.
* D3 is preserved inside the loop, so set it once just before the loop starts.
*-------------------------------------------------------------------------------
            moveq   #-1,d3          ; Timeout for the channel

nl_loop     cmpa.l  bv_nlp(a6),a3   ; Are we done yet ? (Compare offsets)
            bge.s   nl_done         ; Yes
            moveq   #io_sstrg,d0    ; Print some bytes please
            move.b  0(a6,a3.l),d2   ; Counter byte from name list
            ext.w   d2              ; Needs to be word sized for IO_SSTRG
            lea     1(a6,a3.l),a1   ; Start of bytes to print
            adda.w  d2,a3           ; Adjust upwards to end of bytes
            addq.l  #1,a3           ; And point at the next size byte
            trap    #3              ; Print the name (preserves A0, A3 and D3)
            tst.l   d0              ; Ok ?
            bne.s   error_exit      ; Oops - failed

nl_nl       moveq   #io_sbyte,d0    ; Code for 'send one byte'
            moveq   #10,d1          ; Newline character
            trap    #3              ; Print newline (preserves A0, A3 and D3)
            tst.l   d0              ; Ok ?
            bne.s   error_exit      ; Oops - failed

            bra.s   nl_loop         ; Lets go round again !

*-------------------------------------------------------------------------------
* If there is no more to do, return to SuperBasic.
*-------------------------------------------------------------------------------
nl_done     moveq   #0,d0           ; No errors
            rts                     ; Exit to SuperBasic

*-------------------------------------------------------------------------------
* Copy the above code for the CHANNEL_ID subroutine to here as it is required.
* I am not printing it here because it is a simple duplication and we need to
* save paper in the magazine !
*-------------------------------------------------------------------------------
channel_id blah blah blah etc
```

Save the file, assemble, fix typing errors and test - super stuff this eh?

When this procedure runs, you can see all the internal names like PRINT, CLOSE etc and also all your

own stuff like NLIST, GREEN, RED etc and also any filenames that you have used without quotes around them. These are names like anything else.

If you try the following:

```
OPEN_NEW #3,ram1_test
NLIST #3
CLOSE #3
```

then load ram1_test into your editor (or copy to scr_), the last entry in the name list will be ram1_test - because you didn't use quotes. If you now try:

```
OPEN_NEW #3,"ram1_test_again"
NLIST #3
CLOSE #3
```

This time, ram1_test_again will NOT be in the list because it is not a name, simply a string. This routine can be used to get a list of all procedures, functions, names etc that are loaded into your QL.

In the next issue, I will explain the maths stack in detail.

COSMOS

A review by Brian Kemmett

There must be many of you that have looked up at the sky on a crystal clear night whilst out walking the dog, draining the last drops from the bottom of your glass during that late summer's evening Barby, or just plain having that quiet five minutes in the cool evening air, and thought, 'I wonder what that bright star up there is - is it a planet, or a star??, I wonder how far away it could be?'..
Well, "TALENT Computer Systems" (now there's a name from the past!!) came to the rescue with "QL COSMOS" originally written by G.F. Cornwell and first advertised - I believe - in "QL USER" December 1985 for the princely sum of £14.95. Rich Mellor of RWAP software has now updated it and has re - released it for an unbelievable price of just £5.00.

What you get

A single Double Density Disk, which also contains the program manual as a QUILL_doc, and the main program files in "zip" format, so you will need 'unzip', which is not supplied, but is freely available from numerous sources, including the "QLToday" cover disks.

The Program

The program unzipped (cosmos_obj) is some 77k, so there should not be any problems running it on any system. (the program has now been compiled using Qliberator. Rich says it will run under any screen resolution, and under SMSQ/E.. but more of that later). If anyone can remember the microdrive version, then they will recall how painfully slow loading was.. There was some 10 or 11 different files that had to be loaded at a SNNAILLS pace, but now that has all changed. All that is needed is to enter into the boot program where the program files are stored, then EXEC_W the program itself, which loads up in a matter of a second or two. I have a subdirectory on my Hard Disk named win1_COSMOS, therefore:

```
100 dev$='win1_cosmos_':
REM use flp for floppy
drive systems
110 EXEC_W  'win1_cosmos_
cosmos_obj';dev$
```

The program files, apart from the main cosmos_obj,are the startup screen, a stunning _scr depicting the earth as seen from the moon (this is one of the problem areas, this shows great under the normal 512x256 screen, but any extended screen of the AURORA or Q40 forces the picture into the top left hand corner of the screen, not surprising really, but a shame, all the same). The second file contains details of the user's location in Latitude & Longitude, which is ENTERed from the Main Menu, and then may be saved to alleviate having to enter your details at every session, and the third file contains a table of all the stars used by the main program.

The Program in use

On booting the user is presented with the startup screen, (take note of the above comments), and a few seconds later, the main menu appears. There are eight different options:

F1: SELECT VIEW

This gives the user a submenu ranging from a view of the whole sky, the overhead section, or variations of the

compass (N,NE,NW,S,SE,SW, etc.), the Inner Solar System and the Outer Solar Sytem.

F2: CHANGE TIME

The user is prompted to enter the Greenwich Mean Time using the format 00.00 (ie:20.32 for 8.32pm), and then the date, same type of format: 27.3.2000 for 27th March 2000. This is the time and date that all the sky charts will now be based upon until selecting F2 from the main menu and altering it.

F3: CHANGE PLACE

This should be your first option. This actually tells the program at which neck of the woods you are actually gazing up into the sky. Rich gives some useful examples in the Instruction_doc, so if you don't know what your Lat & Long. are, you can choose and use the nearest one to your location. (I got the details of my location from the border of an Ordnance Survey map) You enter the details in the format 48.22n 2.18e, then you are given the option of saving the info for future use.

F4: DIRECT SELECT

This gives another submenu allowing the user to select any of the nine planets, or the sun, moon & stars. For an example, if I select JUPITER, I get all the relevant details, such as location in the sky, its diameter, temperature, rotation period(day) apparent size and number of moons. On pressing any key I am given a graphical representation of how it would appear in a telescope, complete with its moons.

F5: CURSOR SELECT

This is the heart of the system. Say, from the main menu you had selected F1 (SELECT VIEW), and opted for the whole sky. In seconds the program will calculate the positions of all stars and planets visible from your location (as set in option F3) and the time and date selected (set in option F2). They will be plotted onto the monitor/TV as dots of varying colours and size in relation to their apparent size and luminosity. There is also be a legend at the top left of the screen giving your Lat & Long, with the date and time of the plot. In the bottom right corner is the instruction telling you to press ESC to return to the main menu. On returning to the MAIN MENU, press F5, and the star chart will reappear, but overlaid with the instructions for using CURSOR SELECT.. pretty simple really. You just use the cursor keys to place the circular cursor over an object on the star chart, and by pressing F5, you are presented with the data for that object, for example, I place the cursor over an obscure dot on the chart, press F5, and I am given the following info:
object type: single star designation(name): Rho Leonis Constellation: Leo distance: 2500light years
I am also given details of its location, magnitude and colour. You get this information for every star/planet displayed on the chart. If you do not line the cursor up properly, you are given details of the closest object to it.. Unfortunately, this is another place where there appears to be a problem. I have an AURORA system and use SMSQ/E v2.91 .. all works fine and dandy. I also have a Q40, using the latest version of SMSQ/E, v2.97.. not so fine. All works fine up to the CURSOR SELECT instructions, but on trying to use the cursor, the instructions remain on the screen, obstructing the star chart and thus making this option unusable. I would hazard a guess this is an operating system fault, and not the program - but I am no expert. Maybe when the final release of the Q40 colour drivers are here,(and thus, the final version of SMSQ/E) maybe this problem will be sorted.

TAB: SCREEN DUMP

As implied, this gives a dump of the current star chart. You are prompted for the device name, including subdirectory, and the file name you wish to call the dump. Remember, this will be a QL screen dump (512x256 scr,32768 bytes), so you will not be able to view this with any extended screens of the AURORA or Q40.

ESC: REMOVE MENU

This option allows you to view the star chart unhindered.. Pressing ESC again will return the MAIN MENU.

F6: QUIT PROGRAM..

Guess What??

Thats about it.. Is it worth it?? Putting the negative comments I have made to one side, I would whole heartedly say yes. A little gem of a program, very simple in its operation and it does what it is intended to do admirably, and without fuss. It's content is as relevant today as it was way back in 1985.. Congratulations to Rich for resurrecting this little classic from obscurity and returning it to the QL community.

Inside GoldFire - Part 1

"Nasta"

As GoldFire is mentioned several times in this issue, and customers are waiting for it for a long time, Nasta has kindly allowed us to print the current specification of the GoldFire. This will give you an idea how complex the GoldFire is and what it will be. As the description is fairly technical, we only dedeicate a few pages per issue - and hope that by the time we print the final part, GoldFire will be available. Even if you are not technically minded, have a look at the text and you can imagine what GoldFire will be.

1. GOLDFIRE BUS PROTOCOL

Table Of Contents

1.0 The purpose of this document
1.1 Bus electrical definition
1.1.1 Expansion connector
1.1.2 Signal types
1.2 Bus signal Definitions
1.2.1 Power and Ground
1.2.2 Address and Data buses
1.2.3 Bus cycle control signals
1.2.4 Interrupt lines
1.2.5 System or motherboard generated signals
1.2.6 Compatibility signals
1.2.7 Reserved lines
1.2.8 Comparison between GoldFire and original QL bus specification
1.3 Bus protocols
1.3.1 Narrow (8-bit) bus protocols
1.3.2 Wide (32-bit) bus protocol
1.4 Other interfacing considerations
1.4.1 Electrical properties of signals
1.4.2 Timing properties of signals
1.4.3 Functional properties of signals
1.5 Specification changes

1.0 The purpose of this document

This document describes in detail external bus interfacing, protocols and compatibility issues for the GoldFire. The GoldFire bus is based on the QL bus and is compatible with it, but also adds an array of additional features. It also redefines the original QL bus definition taking into account 17 years of QL expansion development.

1.1 Bus electrical definition

GoldFire uses an expanded subset of QL expansion bus signals. The actual connector has remained unchanged, and is compatible with many QL expansion boards both mechanically and within certain limits, electrically.

1.1.1 Expansion connector

The following is the signal layout on the expansion connector:

D-32	A-32	8bit	T	G	CONNECTOR PINOUT		G	T	8bit	D-32	A-32
----<	GROUND	>----	P	P	1b	1a	P	P	----<	GROUND	>----
D26	X	D2	B	B,*	2b	2a	B,*	B	D3	X	D27
D25	A25	D1	B	B,*	3b	3a	B,*	B	D4	BS0	D28
D24	A24	D0	B	B,*	4b	4a	B,*	B	D5	BS1	D29
----<	-WTRQ	>----	U	O,*	5b	5a	B,*	B	D6	BS2	D30
H	H	-NTRQ	B	B,*	6b	6a	B,*	B	D7	BS3	D31
----<	-WR	>----	U	O,*	7b	7a	B,*	B	A19	A19	D19
----<	-TACK	>----	L	I	8b	8a	B,*	B	A18	A18	D18
----<	-WDATA	>----	U	O,*	9b	9a	B,*	B	A17	A17	D17
----<	-ATRQ	>----	U	O,*	10b	10a	B,*	B	A16	A16	D16
D15	A15	A15	B	B,*	11b	11a	-	C,U	----<	CLK	>----
----<	-RESET	>----	S,U	I	12b	12a	-	C,U	----<	-VPA	>----
----<	-RFSH	>----	C,U	-	13b	13a	B,*	B	A14	A14	D14
D23	A23	A23	B	B,*	14b	14a	B,*	B	A13	A13	D13
----<	POLL	>----	S,U	I	15b	15a	B,*	B	A12	A12	D12
----<	RESERVED	>----	R	-	16b	16a	B,*	B	A11	A11	D11
----<	RESERVED	>----	R	-	17b	17a	B,*	B	A10	A10	D10
----<	RESERVED	>----	R	-	18b	18a	B,*	B	A9	A9	D9
D22	A22	A22	B	B,*	19b	19a	B,*	B	A8	A8	D8
D21	A21	A21	B	B,*	20b	20a	B,*	B	A7	A7	D7
D20	A20	A20	B	B,*	21b	21a	B,*	B	A6	A6	D6
D0	X	A0	B	B,*	22b	22a	B,*	B	A5	A5	D5
----<	ROMOE	>----	C,U	-	23b	23a	B,*	B	A4	A4	D4
D1	X	A1	B	B,*	24b	24a	B,*	B	A3	A3	D3
D2	A2	A2	B	B,*	25b	25a	-	R	----<	RESERVED	>----
----<	GROUND	>----	P	P	26b	26a	P	P	----<	GROUND	>----
----<	-HINT	>----	L	-	27b	27a	O,H	H	----<	DDEC8	>----
----<	IPL1	>----	C,L	I	28b	28a	P	P	----<	GROUND	>----
----<	-NINT	>----	L	I	29b	29a	P	P	----<	GROUND	>----
----<	-LINT	>----	L	O,L	30b	30a	P	P	----<	+12V	>----
----<	+5V	>----	P	P	31b	31a	P	P	----<	-12V	>----
----<	+5V	>----	P	P	32b	32a	P	P	----<	+5V	>----

Legend:

TABLE HEADING:
- D-32 32 bit cycle data phase (-WDATA is LOW)
- A-32 32 bit cycle address phase (-WDATA is HIGH)
- 8bit 8-bit cycle
- T Signal type
- G Use on GoldFire

SIGNAL TYPES (Electrical and usage):
- P Power or ground lines
- B Bidirectional lines
- U Unidirectional lines (direction as defined in signal definitions)
- L Pulled low
- H Pulled high
- R Reserved
- S System (see signal description)
- C Compatibility (see signal description)

SIGNAL USE ON GOLDFIRE:
- P Power or ground connection
- B Bidirectional
- O Output
- I Input
- - Not connected
- * 33 ohm series terminated
- L Sink ('open collector')
- H Source ('open emitter')

SIGNAL NAMES AND NOTATIONS:
-name active low signal
-‹name›- non-multiplexed signal
L, H, X Low, High, Unknown or don't care state

1.1.2 Signal types

Signals can be of three types by usage:

1) General interconnect: These signals are regular bus signals and power lines as used by components on the bus.

2) System: These signals are required for proper system operation and are provided by a 'chasis' or a 'motherboard'. There are only two such signals: the -RESET signal, and the POLL signal.

3) Compatibility: These signals are connected on backplanes but are not necessarily used within a system. They are provided for compatibility with existing QL motherboards and peripherals.

Signals can be of 4 types by electrical definition:

1) Bidirectional (fully driven): This implies that a signal may be driven by one output at a time, and used by many inputs. In effect these are 'totem pole' TTL compatible signals, which can be tri-stated, although the high impedance state is not used with the GoldFire other than when a signal is an input or in input mode. On GoldFire, all bidirectional signals save one are bus lines, and they are series terminated with 33 ohm resistors. In addition, the outputs are LVTTL compatible, which means they can interface directly with 3.3V devices. As outputs, these lines can sink or source at least 12mA, and are therefore fully buffered. Backplanes may provide additional parallel termination and level fix (by pull-up or pull-down).

2) Unidirectional signals (fully driven): This implies signals which always have one direction, either input or output. On GoldFire, fast outputs such as bus cycle control signals, are series terminated with 33 ohm resistors, are 3.3V compatible, and can sink or source at least 12mA. Inputs are pulled to a defined state, which is always 'high' on GoldFire.

3) Wired logic signals (pull up or pull down): These are various signals that implement wired OR / wired AND logic, and are either inputs or pull-up or pull down outputs. GoldFire implements only two such signals, DDEC8 and -EINT. Both can source or sink 12mA, respectively. The bus provides for 3 more wired logic signals, but those are inputs to the GoldFire.

4) Power and ground lines: The new bus definition puts 5V power in the place of 9V lines as has become a custom with today's QL configurations. The GoldFire is, however, very flexible, and can use any voltage level between approximately 4.4V and 12V on it's main power pins. This capability is provided for compatibility with QL systems that operate on 9V power. +12V and -12V power is not used directly on the GoldFire, it is routed to the port expansion connector.

1.2 Bus signal Definitions

This section defines the functions of each signal on the bus in detail.

1.2.1 Power and Ground

The bus provides four power nodes:

+5V This is the main power node, it uses three pins for higher current capability. The GoldFire will also accept 9V power on these lines for compatibility with older QL systems without the need for modification. At least 1.8A should be available from this line at 5V (or 1A at 9V) for proper GoldFire operation.

+12V This is the +12V low current utility power node. It is not used by the GoldFire.

-12V This is the -12V low current utility power node. It is not used by the GoldFire.

GND This is the combined ground node for signals and power. It uses 6 pins for higher current capability.

1.2.2 Address and Data buses:

Axx Address, data or multiplexed address and data lines, referred to

Dxx depending on application (32bit or 8bit). For 8-bit non-multiplexed

BSxx cycles the usual A0..A23 and D0..D7 is used to describe the pins. For 32-bit multiplexed cycles A/Dx, Ax/Dx or BSx/Dx is used to describe the pin, by the information it carries in the address/data phase. Refer to 1.5 for possible changes to the multiplexing scheme. When signals are referred to by the address they carry, the DE-MULTIPLEXED definition is always used, i.e. Axx, Dxx, BSx. Byte select signals BS0..BS3 select bus bits D31..D24, D23..D16, D15..D8 and D7..D0 respectively. In effect, the number appended to BS denotes the combination of A0 and A1 which would access bytes in a long word at the address [A25][A24]..[A3][A2]00b. Byte select signals are ACTIVE HIGH.

1.2.3 Bus cycle control signals

-NTRQ Narrow (8 bit) transfer request. This signal goes low when a narrow cycle starts and is held low throughout the duration of the cycle. The signal uses the QL bus DSL pin and is compatible to the QL DSL signal. This signal does NOT go low when the autoterminated narrow IO area is accessed (see signal -ATRQ below).

-ATRQ Auto-terminated narrow cycle request. This signal is similar to -NTRQ except that it is generated only for auto-terminated narrow accesses. These accesses have a fixed duration and cover an address range of 16M bytes. The signal is active throughout the cycle which lasts approximately 190ns. This signal uses the QL bus BGL pin, and it's function is NOT compatible with the QL bus definition. THE 68008 CPU MUST be removed from the system if a standard QL motherboard is used with GoldFire.

-WTRQ Wide (32 bit) transfer request. This signal goes low when a 32bit cycle starts and is held low throughout the duration of the cycle. The signal uses the QL bus BGL pin. Because of this, if the GoldFire is used with a standard QL motherboard, THE 68008 CPU MUST BE REMOVED from it's socket.

-WDATA Wide data phase. When this signal is low, the 32-bit cycle data phase is in progress. The signal uses the QL bus ASL pin and is NOT COMPATIBLE with it's previous function. ASL should not be used on QL peripherals anyway so this should not present a problem.

-WR Write cycle. When low, data is output to the bus when appropriate in the cycle. The signal is used in a similar manner for both the harrow and wide bus mode. In the wide bus mode, the signal goes low on a write cycle in the address phase and remains low throughout the cycle, to facilitate decoding. The signal uses the QL bus RDWL pin and is QL compatible

-TACK Transfer acknowledge. This signal has to be pulled low according to the bus protocol of the bus cycle in progress. The signal uses the QL bus DTACKL pin and is compatible with it when used for narrow cycles. 8 bit QL peripherals will not see wide cycles and will not generate this signal on a wide cycle. -TACK is ignored during an auto-terminated narrow access cycle.

1.2.4 Interrupt lines

-NINT Non-maskable interrupt. This signal uses the QL bus BERRL pin, which is normally used to cause a bus error exception or a cycle re-run. No known QL peripherals generate this signal, hence it was redefined as a non maskable interrupt input, active low, edge triggered. On the GoldFire this pin is an input to the interrupt controller.

-HINT High priority interrupt. This signal uses the QL bus IPL0 pin. This pin is normally not used by peripherals except the IPC. If a standard IPC is used, the CTRL-ALT-5 keypresses will generate a level 4 interrupt to CPU1 on the GoldFire, because this pin is an input to the interrupt routing hardware and maps to interrupt level 4 on CPU1. It is recomended that the IPC pin connected to IPL0 be bent out of the IPC socket.

-LINT Low priority interrupt. This signal uses the QL bus EXTINTL pin and has the same function as the QL signal of that name. On the GoldFire this pin is an INPUT into the interrupt controller. The GoldFire may use this input even when there is no ZX8302 ULA in the system.

1.2.5 System or motherboard generated signals

-RESET Reset signal, goes low on power up or when reset button is pressed. Fully QL compatible, always an input to all boards except the motherboard or system backplane.

-POLL Polling interrupt signal, approximately 50Hz frequency. This signal uses the QL bus VSYNCH pin and is compatible with the QL bus definition. The signal is provided to generate a stable timing reference as expected by QL OS software. On the GoldFire this signal is an input to the interrupt controller.

1.2.6 Compatibility signals

DDEC Decoder disable. This signal uses the QL bus DSMCL pin and is similar in function with the corresponding QL bus signal. When DDEC is pulled high, the 'motherboard' will disable it's internal decoders and not react to any cycle on the

bus. This signal is used to enable additional or different decoding than is provided by the 'motherboard' or to disable motherboard devices on an address by address basis. This is the only compatibility signal generated by the GoldFire, and is pulled high whenever A18 and A19 are either 01 or 10. This prevents potential aliasing of motherboard devices into parts of the narrow address range which could not be used in previous QL expansion boards such as the (Super) Gold Card.

CLK 7.5MHz QL compatible system clock. This signal is normally generated by the 'motherboard', and is very rarely used in the system.

RFSH Refresh clock. This signal uses the QL bus CSYNCHL pin, and normally carries the CSYNCH signal if a QL motherboard is in use, or a constant 16384Hz signal if an Aurora is in use. This signal has been retained for compatibility with the (Super) Gold Card which uses it as a reference for RAM refresh. The GoldFire does not use this signal.

-VPA Valid Peripheral Address. This signal was originally used for 6800 family peripheral compatible cycles, and to force autovectoring of interrupts on standard QLs and the Gold Card. Both the original QL motherboard and the Aurora generate a low on this line when the FC0 and FC1 lines are both high which for the aforementioned hardware signals an interrupt acknowledge cycle. Since these lines have also been redefined into data and address lines, the state of -VPA will be indeterminate when a GoldFire is used in the system, however, the -VPA line is not used in this configuration anyway so this should not be a problem.

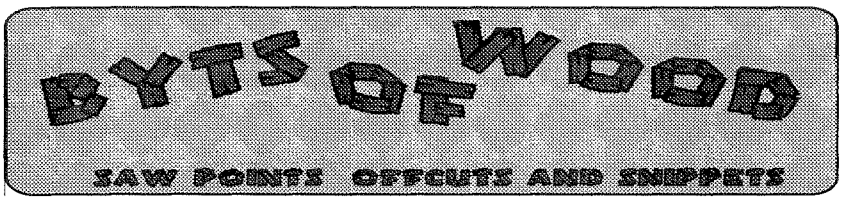
-IPL1 Interrupt input line priority 1. This line is used to generate level 2 interrupts in QL setups. This line is retained for compatibility reasons and is an input on the GoldFire, tied into the interrupt controller to produce level 2 interrupts on CPU1.

ROMOE ROM enable on the QL motherboard. This signal is normally generated by the 'motherboard' whenever the address on the bus is in the range 00000h..0FFFFh. Both the original QL motherboard and the Aurora generate this signal for ROM slot compatibility. The GoldFire doesn't use it.

1.2.7 Reserved lines

R
G
B

The lines which have these designators in the standard QL bus definition are not used by any known peripheral and have also been left unused by the GoldFire. They are considered reserved. It is not guaranteed that DBGL backplanes will have any interconnect on these lines.



The last column was put to bed during the frantic preparations for the annual trip to the U.S. for the show there. Now I am back having had a very enjoyable time with the US QLers. Tony, Jochen and myself stayed at Bill Cable's hillside retreat for the weekend of the show and I, for one, would like to thank Bill for his hospitality.

Hot HTML

A couple of issues back I mentioned that I would like to be able to produce HTML on a QL based machine and, at the U.S. show Francious Lancialt (author of Paragraph the ProWesS Word Processor) demonstrated that he had done that very thing in the current release of Paragraph (2.03). By the time that you read this I hope to have a Paragraph produced page up on my website. The demonstration of the capabilities of Paragraph at the show was really excellent and this will prove to be a very useful tool when the faster machines begin to take hold. Certainly on the Q 40 this is a very efficient tool and now that people are running QPC 2 on ATHLON based systems in excess of 650MHz I am sure they will reap the rewards of his hard work.

'There's GoldFire in Them Thar Hills'

At Bill Cable's Sunday barbeque after the show Nasta and Tony Firshman were deep in conversation about the future of the GoldFire project. Nasta

has been working hard to get this off the ground and they are already into parts selection and other exciting things. Nasta was beginning to talk in terms of what we may have to leave off given the amount of capabilities the chips he wants to use have. The main problem being how do you fit all the connectors on.
This is all good news to those who like to have the feel of the native QL Hardware. The GoldFire is planned to be about the same size as the Aurora Motherboard so it should fit nicely into the MinisQL cases.

Colour Issues.

Marcel Kilgus was kind enough to let me have one of the beta test versions of the new colour driven QPC 2s. I have been running this on my Athlon 700 Mhz PC and I found it very powerful. Although this is not up to the speed of the Q 40 it certainly works well and he has now implemented a version which runs within a standard Windoze window making it easy to resize and control the program. He has increased the speed of the main kernel in the process of writing this new version and there is very little loss of reaction even when running in the highest resolution and colour mode.

At one of our recent user group meetings we tried this out on the various laptop machines our members have. We did note that the machines with the lowest graphics card memory seemed to have the most problems when running the program and Marcel confirms that he expects 4Mb graphics card memory to be the minimum for a stable system. Not surprising really since machines with a lower capacity than this have problems running Windoze in full colour mode at high resolution.
This is a good development for QPC 2 and we should soon see other systems sporting this new multi colour look. QPC 2 v 2 should be available in the autumn.

Don't Try This At Home

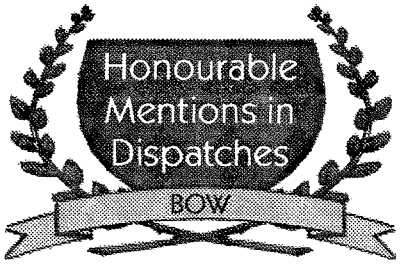
One Q 40 user has reported a problem which occurred with his Q 40 when he swapped monitors while the machine and monitor were both switched on. This caused a spike which, in turn, blew the video output of the Q 40. Although it is tempting to do this when you are playing around with monitors it is something which should be avoided. Most monitors will not do this but there is always a chance of finding one which will or one which has developed a fault allowing high voltages on the input lines. This was not a fatal problem and we fixed it quite quickly but it is not worth the risk.
I would like to point out, while the Q 40 is under the discussion that Dilwyns comments on the reasons, mentioned in the editorial of the last issue, for the supply of Q 40s drying up were not 100% accurate or at least gave a misleading impression. The main reason was that, as Tony Firshman built the boards, he put the ones which failed to one side to be looked at later. I did not know this and carried on selling them until I had sold all of the working boards. At this point Tony should have started to find the problems with the non-working boards. On the whole this is usually the odd bad solder connection or faults of that nature.
Tony, however, had committed himself to other tasks and there was a lull in production. It did not help that he had left himself with a non-functioning processor to do the testing with either.
The other factor which has held the Q 40 up has been something which is odd in the extreme. When I first started using SMSQ/E on the Q 40 I reported all software failures to Tony Tebby. One of the failures revolved around certain QLiberated programs. The quickest test for this was to see if the Lonely Joker would work or not.
At some point in this interchange TT said 'It works on my machine' but it still did not work on mine. He even sent me a disk with a boot which loaded SMSQ/E, loaded QPAC 2 and ran the Lonely Joker. This failed on my one too. I wondered if there was some difference between his, pre-production, board and ours until, at the show in Cambridge last year Tony Firshman ran the disk on his machine and it worked. Since then I have tried everything to pin down the difference between my Q 40 and the others. I even had a customer's Q 40, which I had just built, and my one open side by side and changed every plug-in chip on the boards to see which one was causing the problems but my machine stubbornly refuses to play the game.
When I work out what is causing this I will let you all into the secret.

Possible SMSQ bug?

Talking of problems with systems prompts me to ask those of you out there who have v2.97 or later of SMSQ/E if you have had any trouble formatting or writing to floppies. I am still running my two main

Aurora systems on v2.95 because it seems unable to format DD disks. I had thought it was a dodgy disk drive and changed it but the problem persists. If anyone else has had this problem please let me know because I need to get to the bottom of the problem.

I should imagine that most SMSQ/E users are currently using HD or ED disks and have not noticed the problem but I have to supply a lot of my software on DDs and so I noticed it quite quickly.



This issue the accolade goes to Norman Dunbar.

The QL Show Agenda

Italian Meeting Sun., 8th of Oct.

Sala Congressi Circoscrizione 2
Via Fratelli Cervi 70
Pieve Modolena
Reggio Emilia

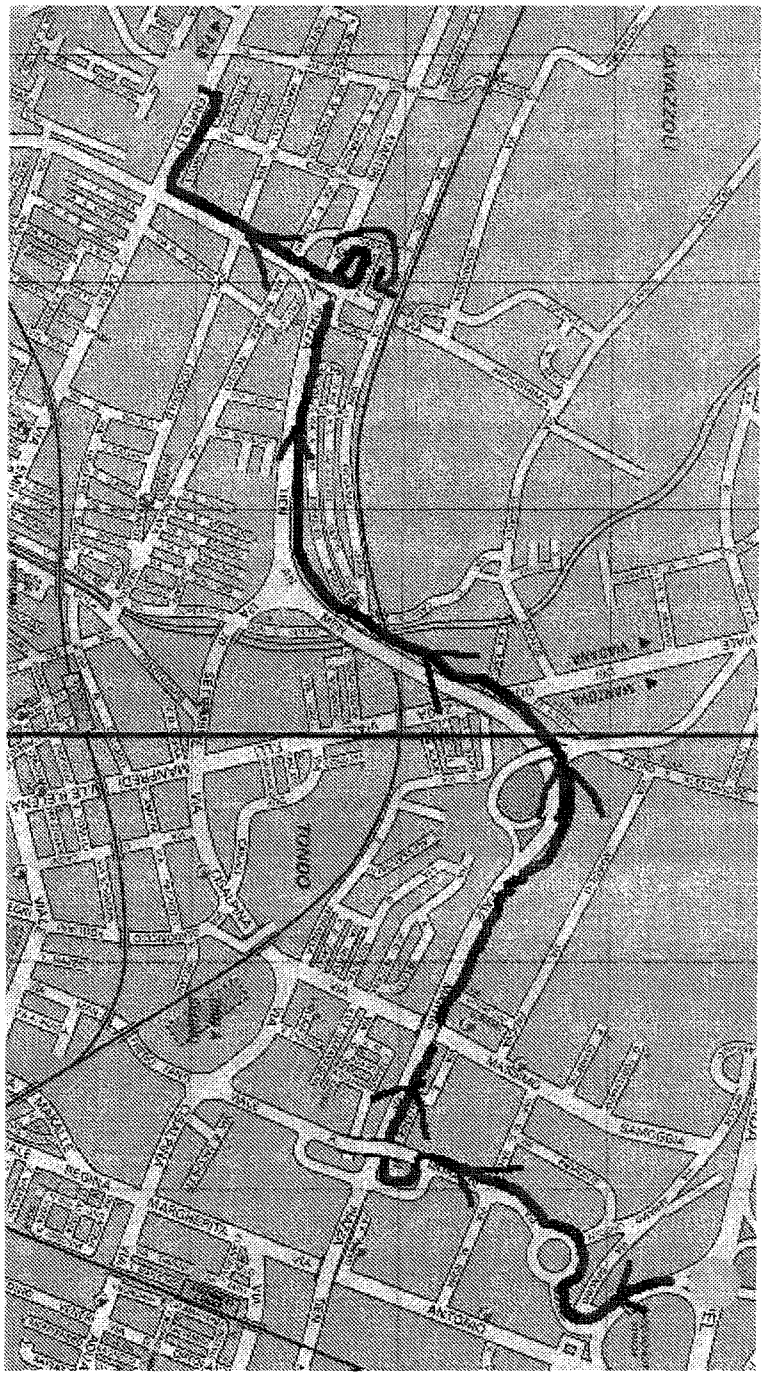
If you come by car, take motorway A1 and exit "Reggio Emilia", then follow the route on the map.

You should be able to find more details about the show soon on Davide's website:

www.geocities.com/dsantachiara/meetinginfoeng.htm

If you need more details, you can contact Davide via Email:
ergon@libero.it

The Italians claimed on the last invitation "come to where you get the best food in the world". So come along to the North of Italy, check out if this is true and join other QL users and dealers. J-M-S will be there (by car), and TF Services and QBranch will try to fly there.



The QL Show Agenda

We will have a hot autumn this year, lots of shows:

Sat., 26th of August - NL-EINDHOVEN!!!

Same venue as always-St. Joris College

Sat., 2nd of September - GB-LONDON

Quanta meeting and trader's show at the
Hammerton Memorial Hall
Lingham Street, Stockwell Green, London SW9 9HF
The venue is approximately 10 minutes walking distance from the following stations:

Brixton Underground (Victoria Line)
Brixton Railway (Orpington to Victoria via Brixton, Herne Hill, West Dulwich, Sydenham Hill, Penge East, Kent House, Beckenham Junction, Bromley South, Bickley and Petts Woods)
Loughborough Junction Railway (Thameslink to Luton and Gatwick Airports)
Clapham North Underground (Northern Line)
Clapham High Street Railway (Victoria to Battersea Park and South East London)
Stockwell Underground (Northern and Victoria Lines)
Overseas vistors travelling via Eurostar will be pleased to note that Stockwell is 3 stops from Waterloo Underground Station on the Northern Line.
Roads - close to the A3, A23 and A24. Look for Stockwell Road on maps and locate Lingham Street SW9.
Buses - several bus routes with destination Stockwell or nearby.
Car parking - There are no parking restrictions after 5.30pm on Friday evening. However, visitors coming by car should try to get here early. Saturday is a shopping day, and you may have to park in nearby side streets. There is parking for about 13 cars at the Hall itself, although do not assume all will be available for our use. This should suit traders to park and off load.

Sat., 30th of Sept. & Sunday, 1st of October - A-Heidenreichstein

Third Austrian Meeting at Heidenreichstein (same venue as before). Details in German issue of QL Today.

Sun., 8th of October - I-Reggio Emilia

Another long awaited meeting in Italy. Same venue as last time, but as this is quite some time ago, you will find the details on the reverse side

Sat., 14th of Oct. and Sun., 15th of October - GB-Portsmouht - QL 2000!

For more details see previous issue of QL Today!

Sat., 21st of October - F-Paris

10:00 to 16:00 (open 9:00 for traders)
Université Paris, 82 Rue de la Liberté, Saint Denis
Room number to be announced. Cafeteria open 11:30 - 15:30

Métro - Ligne 13, terminus Saint-Denis - Université. Bus - lignes 253, 254, 255, 260, 268, 356, 361
http://www.univ-paris8.fr/up8/moyens_d_acces.shtml